SUPRA SERVER PDM

Utilities Reference Manual
(UNIX & VMS)

P25-6220-47

SUPRA® Server Utilities Reference Manual (UNIX & VMS)

Publication Number P25-6220-47

Cincom Systems, Inc.
55 Merchant Street
Cincinnati, Ohio 45246-3732
U.S.A.

PHONE: (513) 612-2300
FAX: (513) 612-2000
WORLD WIDE WEB: http://www.cincom.com

Attention:

Some Cincom products, programs, or services referred to in this publication may not be available in all countries in which Cincom does business. Additionally, some Cincom products, programs, or services may not be available for all operating systems or all product releases. Contact your Cincom representative to be certain the items are available to you.

Release information for this manual

The *SUPRA Server PDM Utilities Reference Manual (UNIX & VMS)*, P25-6220-47, is dated January 15, 2002. This document supports Release 1.3 of SUPRA® Server under UNIX and Release 2.4 of SUPRA Server under VMS environments.

We welcome your comments

We encourage critiques concerning the technical content and organization of this manual. Please take the survey provided with the online documentation at your convenience.

Cincom Technical Support for SUPRA Server PDM

FAX: (513) 612-2000
Attn: SUPRA Server Support

E-mail: helpna@cincom.com

Phone: 1-800-727-3525

Mail: Cincom Systems, Inc.
Attn: SUPRA Server Support
55 Merchant Street
Cincinnati, OH 45246-3732
U.S.A.

Contents

About this book ix

Introduction to utilities 17

Using DBA utilities 31

Using the Database Verify utility 263

Index 287

About this book

Using this document

The *SUPRA Server PDM Utilities Reference Manual* provides instructions for:

- Using DBA utilities to change the physical and logical structure of a database without losing previously stored data.

- Using Fast utilities to change the physical structure of a live database, or the logical record length, without losing previously stored data.

- VMS Using the Batch Directory Maintenance utility (DIRM) to unload the database definition, logical views, and user-to-view permissions from one Directory and load them onto another Directory.

- Using the Database Verify utility to: (1) check the integrity of data sets by comparing their actual physical characteristics with the information stored in the database description file and the SUPRAD Directory, and (2) gather physical statistics about data sets.

Document organization

The information in this manual is organized as follows:

Chapter 1—Introduction to utilities

Describes the four sets of utilities: DBA utilities, Fast utilities, Batch Directory Maintenance utility (VMS), and Database Verify utility.

Chapter 2—Using DBA utilities

Describes the DBA utilities—database administration functions available from a menu-driven tool called SUPRA DBA.

Chapter 3—Using Fast utilities

Describes how to use Fast utilities to calculate disk space requirements and make various physical modifications.

Chapter 4—Using SUPRA PDM Batch Directory Maintenance (DIRM) (VMS)

Describes Batch Directory Maintenance (DIRM), an alternative to the DBA utility for the VMS environment. DIRM uses a batch script file with a specific format to export and import metadata for databases.

Chapter 5—Using the Database Verify utility

Describes how to use the Database Verify utility to verify a data set's physical record locations and record pointers, verify a data set's physical characteristics with the information in the database description file and the information on the SUPRAD Directory, and gather physical statistics about a data set.

Index

Revisions to this manual

The following changes have been made for this release:

- A note was added to “Using the Add utility” on page 106 and “Using the Delete utility” on page 115.
- More information was added to the UNIX note under “Calculating disk space requirements” on page 133.
- A note discussing the terms “change file” and “verify list file” was added on page 278.
- Information was added to the /CHANGE_LIST parameter on page 281.
- A UNIX note was added to item 2 under “Using the Database Verify utility with a verify list file” on page 282.

Conventions

The following table describes the conventions used in this document series:

Convention	Description	Example

`Constant width type`	Represents screen images and segments of code.	`PUT 'customer.dat'`
		`GET 'miller\customer.dat'`
		`PUT '\DEV\RMT0'`

Slashed b (b̸)	Indicates a space (blank). The example indicates that four spaces appear between the keywords.	`BEGNb̸b̸b̸b̸SERIAL`
Brackets [ ]	Indicate optional selection of parameters. (Do not attempt to enter brackets or to stack parameters.) Brackets indicate one of the following situations:	
	A single item enclosed by brackets indicates that the item is optional and can be omitted. The example indicates that you can optionally enter a WHERE clause.	[WHERE *search-condition*]
	Stacked items enclosed by brackets represent optional alternatives, one of which can be selected. The example indicates that you can optionally enter either WAIT or NOWAIT. (WAIT is underlined to signify that it is the default.)	$\begin{bmatrix}\underline{\textbf{(WAIT)}}\\ \textbf{(NOWAIT)}\end{bmatrix}$

Convention	Description	Example
Braces { }	Indicate selection of parameters. (Do not attempt to enter braces or to stack parameters.) Braces surrounding stacked items represent alternatives, one of which you must select. The example indicates that you must enter ON or OFF when using the MONITOR statement.	**MONITOR** $\begin{Bmatrix}\textbf{ON}\\\textbf{OFF}\end{Bmatrix}$
Underlining (In syntax)	Indicates the default value supplied when you omit a parameter. The example indicates that if you do not choose a parameter, the system defaults to WAIT.	$\begin{bmatrix}\underline{\textbf{(WAIT)}}\\\textbf{(NOWAIT)}\end{bmatrix}$
	Underlining also indicates an allowable abbreviation or the shortest truncation allowed. The example indicates that you can enter either STAT or STATISTICS.	$\underline{\texttt{STAT}}\texttt{ISTICS}$
Ellipsis points...	Indicate that the preceding item can be repeated. The example indicates that you can enter multiple host variables and associated indicator variables.	`INTO :`*`host-variable`*` [:`*`ind-variable`*`],...`

Convention	Description	Example

UPPERCASE
lowercase
	In most operating environments, keywords are not case-sensitive, and they are represented in uppercase. You can enter them in either uppercase or lowercase.	`COPY MY_DATA.SEQ`
		`HOLD_DATA.SEQ`

	In the UNIX operating environment, keywords are case-sensitive, and you must enter them exactly as shown.	`cp *.QAR /backup`
Italics	Indicate variables you replace with a value, a column name, a file name, and so on. The example indicates that you must substitute the name of a table.	`FROM` *`table-name`*

Punctuation marks	Indicate required syntax that you must code exactly as presented.
	`( )` parentheses
	`.` period
	`,` comma
	`:` colon
	`' '` single quotation marks
		`(`*`user-id`*`,` *`password`*`,` *`db-name`*`)`
		`INFILE 'Cust.Memo' CONTROL LEN4`

SMALL CAPS	Represent a keystroke. Multiple keystrokes are hyphenated.	ALT-TAB
UNIX **VMS**	Information specific to a certain operating system is flagged by a symbol in a shadowed box (**UNIX**) indicating which operating system is being discussed. Skip any information that does not pertain to your environment.	**UNIX** To delete these files, return to the shell and use the rm command. **VMS** To delete these files, return to the command level and use the DELETE command.

SUPRA Server documentation series

SUPRA Server is the advanced relational database management system for high-volume, update-oriented production processing. A number of tools are available with SUPRA Server including DBA Functions, DBAID, precompilers, SPECTRA, and MANTIS. The following list shows the manuals and tools used to fulfill the data management and retrieval requirements for various tasks. Some of these tools are optional. Therefore, you may not have all the manuals listed. For a brief synopsis of each manual, refer to the *SUPRA Server PDM Digest for VMS Systems*, P25-9062.

Overview

- *SUPRA Server PDM Digest for VMS Systems*, P25-9062

Getting started

- *SUPRA Server PDM UNIX Installation Guide*, P25-1008
- *SUPRA Server PDM VMS Installation Guide*, P25-0147
- *SUPRA Server PDM UNIX Tutorial*, T25-2262
- *SUPRA Server PDM VMS Tutorial*, T25-2263

General use

- *SUPRA Server PDM Glossary*, P26-0675
- *SUPRA Server PDM Messages and Codes Reference Manual (PDM/RDM Support for UNIX & VMS)*, P25-0022

Database administration tasks

- *SUPRA Server PDM Database Administration Guide (UNIX & VMS)*, P25-2260
- *SUPRA Server PDM System Administration Guide (VMS)*, P25-0130
- *SUPRA Server PDM System Administration Guide (UNIX)*, P25-0132*
- *SUPRA Server PDM Utilities Reference Manual (UNIX & VMS)*, P25-6220

- *SUPRA Server PDM Directory Views (VMS)*, P25-1120
- *SUPRA Server PDM Windows Client Support User's Guide*, P26-7500*
- *SPECTRA Administrator's Guide*, P26-9220**

Application programming tasks

- *SUPRA Server PDM Programming Guide (UNIX & VMS)*, P25-0240
- *SUPRA Server PDM System Administration Guide (VMS)*, P25-0130
- *SUPRA Server PDM System Administration Guide (UNIX)*, P25-0132*
- *SUPRA Server PDM RDM Administration Guide (VMS)*, P25-8220
- *SUPRA Server PDM Windows Client Support User's Guide*, P26-7500*
- *MANTIS Planning Guide*, P25-1315**

Report tasks

- *SPECTRA User's Guide*, P26-9561**

Manuals marked with an asterisk (*) are listed twice because you use them for different tasks.

Educational material is available from your regional Cincom education department.

1

Introduction to utilities

SUPRA provides four sets of utilities:

- DBA utilities (see “SUPRA DBA utilities” on page 19)
- Fast utilities (see “Fast utilities” on page 23)
- Batch Directory Maintenance utility (VMS) (see “Batch Directory Maintenance utility (VMS)” on page 28)
- Database Verify utility (see “Database Verify utility” on page 29)

The DBA utilities allow you to change the physical and logical structure of a database without losing previously stored data. You can change both the physical attributes of a database definition such as file size, load limit, total number of records per block, as well as logical attributes such as data item length and record layout. In addition, you can add and delete lists of records, as specified in an input file, and produce database statistics. You invoke DBA utilities from the Administration Functions menu and specify the utility functions in response to a series of prompts.

The Fast utilities perform a subset of the DBA utilities. They allow you to change the physical structure of your database without losing previously stored data.

VMS

Because of the DCL interface, Fast utilities execute more quickly than DBA utilities. To invoke Fast utilities from the command level, enter CHANGEDB with parameters to specify the changes to the database.

UNIX

Because of the shell interface, Fast utilities execute more quickly than DBA utilities. To invoke Fast utilities from the shell, enter the command csmchangedb together with parameters to specify the changes to the database.

VMS

The Batch Directory Maintenance utility (DIRM) allows you to unload the database definition, logical views, and user-to-view permissions from one Directory and load them onto another Directory. The transferred definition includes recovery logs, file specifications, base and derived views, domains, and validation tables, but it excludes user-entered data.

The Database Verify utility allows you to verify the integrity of your data sets by comparing a data set’s physical characteristics with the information in the database description file and the SUPRAD Directory. You can also use the Database Verify utility to generate a report on a data set’s physical statistics.

SUPRA DBA utilities

SUPRA DBA allows you to perform the following database administration functions:

- Unload or Reload a primary data set
- Unload or Reload a related data set
- Add records to a data set
- Delete records from a data set
- Produce statistics

You access these functions through a menu-driven tool called SUPRA DBA. See “Using DBA utilities” on page 31 for information about how to sign on to SUPRA DBA and initiate DBA utilities. The following table lists each DBA utility and briefly describes its application.

NOTE

Fast utilities, which execute as a background task, is an alternative method of performing DBA utilities. Fast utilities executes more quickly than DBA utilities; however, it requires more resources to run. Therefore, before using Fast utilities, ensure that you have sufficient resources (see “Calculating disk space requirements” on page 133).

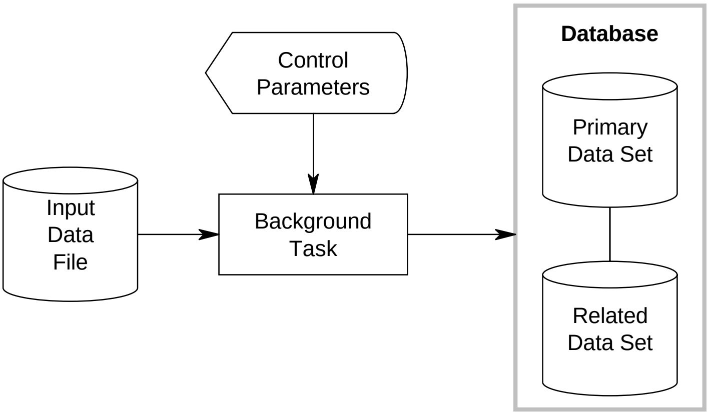
Utility	Application
UNLOAD/ RELOAD PRIMARY DATA SET	Expand the physical characteristics of a data set. Add and remove data items and linkpaths. (You should first create a target database for the new definitions.) VMS Perform a dynamic populate when reloading a data set that has one or more indices.
UNLOAD/ RELOAD RELATED DATA SET	Expand the physical characteristics of a data set. Add and remove data items. (You should first create a target database for the new definitions.) VMS Perform a dynamic populate when reloading a data set that has one or more indices.
ADD RECORDS TO DATA SET	Add records to an existing or new data set. (You should first create a data file containing the data for the records to be added.)
DELETE RECORDS FROM DATA SET	Delete records from a data set. (You should first create a data file containing a list of the keys of the records to be deleted.)
PRODUCE STATISTICS	Produce a statistics report on the specified database. The report can include data set details and performance statistics.

NOTE

You can also use DBA utilities to modify coded records in a related data set.

Once you select Utilities from SUPRA DBA, an interactive preprocessor and background task perform the functions listed in the table above. The preprocessor prompts you to define the utility functions, then uses these definitions to build parameter files containing control information used by the background task.

For the add record and delete record functions, you create a sequential input data file containing the records to be processed by the utility. The input data file must be available before you run the preprocessor. The following figure is a simplified illustration of utility processing:



Executing DBA utilities

You can execute utilities in the following different ways:

- Execute them immediately from DBA after you have responded to the prompts
- Save your responses and execute them later at the command level (VMS) or the shell (UNIX)
- Save your responses and execute them later from DBA

For more information on executing utilities, see "Executing DBA utilities" on page 75.

Modifying SUPRAD using DBA utilities

You cannot modify the Directory database, SUPRAD, if you are signed on to it. Therefore, to run FORMAT, RECOVERY or UTILITIES against the Directory database, you must sign on to DBA with the user name UTILITIES. Refer to the *SUPRA Server PDM System Administration Guide (VMS)*, P25-0130, or the *SUPRA Server PDM System Administration Guide (UNIX)*, P25-0132, for a discussion of the FORMAT and RECOVERY functions when used against the Directory.

The user name UTILITIES does not sign on to the Directory, but it allows you to select the DBA utilities option from the DBA Function Selection menu. The user name UTILITIES is not stored on the Directory, and you cannot change its password.

Fast utilities

Fast utilities operate at the command level (VMS) or the shell (UNIX) and offer an alternative to the DBA unload and reload utilities.

To initiate Fast utilities, enter the command CHANGEDB (VMS) or csmchangedb (UNIX) together with a list of parameters. Alternatively, you can create a command file or a change file containing values for the parameters, and run Fast utilities using the file.

The following table lists the Fast utilities' parameters. See "Using Fast utilities" on page 127 for detailed information about the Fast utilities' commands and parameters.

Parameter	**Function**
/DATASET=*data-set-name* **VMS** DATASET=*data-set-name* **UNIX**	Specifies the name of the data set to be processed.
/data-set-type **VMS** *data-set-type* **UNIX**	Indicates the data set type (primary or related).
/TOTAL_RECORDS=*new-file-size* **VMS** TOTAL_RECORDS=*new-file-size* **UNIX**	Changes the physical size of a data set to accommodate the number of records specified. The size is rounded up to a multiple of the records per block or, for related data sets, to a multiple of the control interval.
/RECORDS_PER_BLOCK=*records* **VMS** RECORDS_PER_BLOCK=*records* **UNIX**	Changes the number of logical records per block.
/CONTROL_INTERVAL=*records* **VMS** CONTROL_INTERVAL=*records* **UNIX**	Related data sets only. Changes the number of logical records that can be held in an internal space-management unit. This value is rounded up to a multiple of the records per block.
/LOAD_LIMIT=*percentage* **VMS** LOAD_LIMIT=*percentage* **UNIX**	Related data sets only. Changes the load capacity for a control interval. This value is specified as a percentage in the range 1–99 (VMS) or 0–99 (UNIX).
/LOGICAL_RECORD_LENGTH=*nnnn* **VMS** LOGICAL_RECORD_LENGTH=*nnnn* **UNIX**	Increases the logical record length.

Parameter	Function
/LINK=*(lkpath1,lkpath2...)* VMS LINK=*(lkpath1,lkpath2...)* UNIX	Related data sets only. Determines the primary linkpath by placing it first in the list of linkpaths.
/ALLOCATION=*(na1,na2...)* VMS ALLOCATION=*(na1,na2...)* UNIX	Specifies relative size of physical output files for storage of logical records. Only needed if the data set is to be held on more than one physical file.
/FILE_SPEC=*(file1, file2...)* VMS FILE_SPEC=*(file1, file2...)* UNIX	Specifies physical disk files assigned to hold part of the data set.
/SHADOW=*(sfile1,sfile2...)* VMS SHADOW=*(sfile1,sfile2...)* UNIX	Specifies the physical disk files to shadow the output file(s).
LOAD[=*file*] UNIX	Indicates that changedb will use an unload (sequential) file as input rather than a data set.
UNLOAD[=*file*] UNIX	Indicates that changedb will unload the data set to an unload (sequential) file.
database-name VMS DB_NAME=*database-name* UNIX	Specifies the name of the database description on the Directory.
/USERNAME=*dba-username* VMS USERNAME=*dba-username* UNIX	Specifies the DBA user name that controls access to the Directory and enables the Change Database utility to access Batch Validate and Compile to create the new database description.
/PASSWORD=*dba-password* VMS PASSWORD=*dba-password* UNIX	Specifies the DBA password for the DBA user name. The Fast utilities prompt for the user password if you do not supply a /PASSWORD parameter (slash not required for UNIX) and the password is nonblank.
/SIGNON_DB_NAME=*db-name* VMS SIGNON_DB_NAME=*db-name* UNIX	Specifies the name used by application programs to sign on to the database.
/DB_PASSWORD=*password* VMS DB_PASSWORD=*password* UNIX	Specifies the database password. The Fast utilities prompt for the database password if you do not supply a /DB_PASSWORD parameter (slash not required for UNIX) and the password is nonblank.

Parameter	Function
/OUTPUT=*target-db-name* **VMS** OUTPUT=*target-db-name* **UNIX**	Identifies the file specification for the compiled target-database description if it is to differ from the input database description.
/NOLIST **VMS** NOLIST **UNIX**	Inhibits the production of a database print file (*database-name*.LIS) after database compilation.
/CHANGE_LIST=*change-file* **VMS** CHANGE_LIST=*change-file* **UNIX**	Specifies a file that contains a list of changes to be applied to one or more data sets.
/NOPOPULATE **VMS** NOPOPULATE **UNIX**	Inhibits automatic population of index files after the change database run.

Processing with Fast utilities

You can use the following methods to run Fast utilities:

Method	See
List the Fast utilities parameters in a "change file"	"Executing Fast utilities using a change file" on page 156
Enter the Fast utilities parameters interactively at the shell (UNIX) or command level (VMS)	"Executing Fast utilities interactively" on page 158
Spawn a Fast utilities subprocess (VMS) or execute it as a background process (UNIX)	"Spawning Fast utilities as a subprocess (VMS) or executing it as a background task (UNIX)" on page 161
List the parameters in a command file	"Executing a Fast utilities command file (VMS) or script (UNIX)" on page 162

For command format and examples, see the appropriate section.

Fast utilities considerations

- When Fast utilities execute, they check whether the specified database is active, and return an error status if any users are still signed on.
- If you do not supply all the necessary parameters, Fast utilities get them from the compiled database description. If you omit the database password, the program prompts for it and exits if you do not enter a valid password. Fast utilities also check all details against the database details held on the Directory.
- You do not need to specify or create a target database description. Fast utilities use Batch Validate and Compile to create the new database description automatically. However, to hold the new compiled database description on a different file from the source database description, you must specify the new file specification through the /OUTPUT= parameter (no slash required under UNIX).
- Fast utilities produce a print file automatically unless you specify the /NOLIST parameter (no slash required under UNIX).
- Fast utilities collect all informational messages issued during a run and deliver them at the end of processing. These messages do not indicate that the run failed; they are generated to make you aware of issues that may require your evaluation.

Modifying SUPRAD using Fast utilities

To modify the Directory database, SUPRAD, you must specify the user name DATA-DICTIONARY and the appropriate password with the /USERNAME parameter.

NOTE

Back up the Directory *before* you attempt to modify it using CHANGEDB (VMS) or csmchangedb (UNIX).

Batch Directory Maintenance utility (VMS)

Batch Directory Maintenance (DIRM) is an alternative to the DBA utility for the VMS environment. DIRM uses a batch script file with a specific format to export and import metadata for databases. The utility is interactive, menu-driven, and provides the following functions:

- **DIRM_LOAD**. Allows you to process existing Batch DIRM input files.
- **DIRM_UNLOAD**. Allows you to unload the metadata for an existing database description or data set from the Directory into a text file containing Batch DIRM input statements. This allows you to apply your dictionary updates online using the DBA utility, followed by a DIRM_UNLOAD to have a Batch DIRM copy of your updates for later use.
- **EXPORT_VIEWS**. Allows you to create a DBAID DCL script file useful for recreating RDM view definitions.
- **VIEWS_FOR_USER**. Allows you to create a DBAID DCL script file useful for recreating RDM view permissions.

A Batch DIRM statement consists of a database entity category (BUFF for buffer), followed by a command stating how that entity will be affected (ADD), followed by one or more parameters describing which part(s) of the entity the command will act upon (NAME, TYPE). The categories that can be acted upon with Batch DIRM statements include the following:

Buffers

Comments for the previous object

Database descriptions

Physical data items

Domains

Data sets

Physical file attributes

Indices

Logical data items

Record codes

RMS keys

Secondary keys

System logs

Task logs

SUPRA user names

Validation tables

For details on the Batch Directory Maintenance utility, see “Using SUPRA PDM Batch Directory Maintenance (DIRM) (VMS)” on page 165.

Database Verify utility

Use the Database Verify utility to:

- Verify a data set's physical record locations and record pointers
- Compare a data set's physical characteristics against the information in the database description file and the information on the SUPRAD Directory
- Gather physical statistics about a data set

The following table shows the Database Verify utility parameters:

Parameter	Function
database-name **VMS** DB_NAME=*database-name* **UNIX**	Specifies the name of the compiled database description that contains the data set to be verified.
/PASSWORD=*database-password* **VMS** PASSWORD=*database-password* **UNIX**	Specifies the password assigned to the database.
/DATASET_NAME=*data-set-name* **VMS** DATASET_NAME=*data-set-name* **UNIX**	Specifies the name of the data set to be verified.
/STATISTICS **VMS** STATISTICS **UNIX**	Specifies whether or not you want statistics to be compiled for the data set while it is being verified.
/DIRECTORY= *option* **VMS** DIRECTORY= *option* **UNIX**	Specifies whether you want the data set information in the SUPRAD Directory validated against the information in the database description file.
/CHANGE_LIST= *change-file* **VMS** VERIFY_LIST= *change-file* **UNIX**	Specifies a file with which you can verify more than one data set at a time.

You can use the following methods to run the Database Verify utility:

Method	See
Verify more than one data set at a time by using a change file.	"Using the Database Verify utility with a verify list file" on page 282
Enter the parameters interactively at the command level (VMS) or the shell (UNIX).	"Executing the Database Verify utility interactively" on page 283
Spawn the Database Verify utility as a subprocess (VMS) or execute it as a background process (UNIX).	"Spawning the Database Verify utility as a subprocess (VMS) or as a background process (UNIX)" on page 284
Execute a Database Verify utility command file (VMS) or script (UNIX).	"Executing a Database Verify command file (VMS) or script (UNIX)" on page 285

2

Using DBA utilities

The DBA utilities are database administration functions available from a menu-driven tool called SUPRA DBA.

NOTE

Fast utilities, which execute as a background task, are another method of performing database administration functions. Fast utilities execute more quickly than DBA utilities; however, they require more resources to run. Therefore, before using Fast utilities, ensure that you have sufficient resources (see “Calculating disk space requirements” on page 133).

What to do before using the DBA utilities

Before you use the DBA utilities, you must:

- If the target database for the Unload/Reload utility is a newly created database, define two logical names: one for the database description file and one for the preferred machine list

VMS

These logicals are normally defined automatically in the PDM_LOGICALS_*XXXXXX*.COM procedure. For a detailed description of this procedure, refer to the *SUPRA Server PDM System Administration Guide (VMS)*, P25-0130.

- Understand which directory the utilities operate in
- Access DBA utilities through menu-driven SUPRA DBA
- Initiate DBA utilities

Defining a logical name for the compiled database description file

If the target database for the Unload or Reload utility is a newly created database, you must define a logical name pointing to the compiled database description file.

UNIX To define the logical name for the compiled database description under UNIX, use the csideflog utility as follows:

```
          ┌-p┐
csideflog │-g│ [xxx_ ]dbname  / path / dbname. mod
          └-s┘
```

```
┌-p┐
│-g│
└-s┘
```

Description *Optional*. Specifies the level of the logical name.

Options -p   Process

-g   Group

-s   System

[*xxx*_]*dbname /path/dbname.mod*

Description *Required*. Assigns a logical name to the compiled database description file.

Format

[*xxx*] *Optional*. 3-character database prefix.

dbname 6-alphanumeric-character database name; first character must be alphabetic.

/*path*/*dbname*.*mod* UNIX file path for the compiled database description file.

Considerations

- Define a separate logical name for each database you want to load.
- You must define this logical name before you can format the physical files for the database.
- Groupwide databases may be accessed only by users in the same group as the initiating task.
- Systemwide databases may be accessed by all users of the system.

VMS

To define the logical name for the compiled database description under VMS, use the DEFINE command as follows:

$$\textbf{DEFINE}\begin{bmatrix}\textbf{/GROUP}\\ \textbf{/SYSTEM}\\ \textbf{/TABLE}=\textbf{CSI\_PDM\_}\boldsymbol{pdmname}\end{bmatrix}$$

$$\textbf{[}\boldsymbol{xxx}\textbf{\_]}\boldsymbol{dbname}\ \boldsymbol{dev}\textbf{:[}\boldsymbol{dir}\textbf{]}\boldsymbol{dbname}\textbf{.MOD}$$

$$\begin{bmatrix}\textbf{/GROUP}\\ \textbf{/SYSTEM}\\ \textbf{/TABLE}=\textbf{CSI\_PDM\_}\boldsymbol{pdmname}\end{bmatrix}$$

Description *Optional*. Specifies the level of the logical name.

Options

/GROUP	Groupwide
/SYSTEM	Systemwide
/TABLE=CSI_PDM_*pdmname*	User logical name table for a multiple systemwide PDM.

[*xxx*_]*dbname dev*:[*dir*]*dbname*.MOD

Description *Required*. Assigns a logical name to the compiled database description file.

Format

[*xxx*_]	*Optional*. 3-character database prefix.
dbname	6-alphanumeric-character database name; the first character must be alphabetic.
dev:[*dir*]*dbname*.MOD	VMS file specification for the compiled database description file.

Considerations

- Define a separate logical name for each database you want to load.
- You must define this logical name before you can format the physical files for the database.
- Groupwide databases may be accessed only by users in the same group as the initiating task.
- Systemwide databases may be accessed by all users of the system.

Defining a logical name for the preferred machine list

If the target database for the Unload/Reload utility is a newly created database, you must define a logical name for the preferred machine list.

UNIX

To define the logical name for the preferred machine list under UNIX, use the csideflog command as follows:

```
           ┌-p┐
csideflog  │-g│ [xxx_]dbname_CSI_PDM_MACS mac1 [,mac2...n]
           └-s┘
```

```
┌-p┐
│-g│
└-s┘
```

Description *Optional*. Specifies the level of the logical name.

Options

-p Process

-g Group

-s System

[*xxx*_]*dbname*_CSI_PDM_MACS *mac1*[,*mac2...n*]

Description *Required*. Identifies a list of machines that can be used on the specified database, in order of preference (the preferred machine list).

Format

[xxx] *Optional*. 1- to 3-character database prefix.

dbname 6-character database name forming the variable part of the logical name. The first character must be alphabetic.

mac1 Node name of the first choice machine on which the database can be used.

[,mac2...n] Node names of any other machines on which the database can be used in order of preference.

Considerations

- Create a preferred machine list for each database to be used.
- If the preferred machine list contains more than one machine, then duplicate the logical assignments on each machine that might execute an application.
- The logical name *dbname*_CSI_PDM_MACS must be accessible to all applications that might access the specified database.
- You may prefer to make these logical assignments elsewhere, either manually or at system startup. In this case, they are not needed in the PDM initiator script. Because these logical names must exist somewhere on the system, you may prefer to duplicate them in your PDM initiator script rather than to risk omitting them.
- The PDM can load a database and access its files if they reside on a different node, provided they are accessible to it (on an NFS mounted file system.) Files made available this way, however, cannot reside in a SUPRA file system and can only be accessed using standard UNIX I/O functions. NFS file access may not be reliable, so it is advisable to keep everything local where possible.
- The 1- to 3-character prefix allows you to specify a different preferred machine list for databases with the same name in the same PDM.

VMS To define the logical name for the preferred machine list, use the DEFINE command as follows:

$$\textbf{DEFINE}\begin{bmatrix}\textbf{/GROUP}\\ \textbf{/SYSTEM}\\ \textbf{/TABLE = CSI\_PDM\_}\textbf{\textit{pdmname}}\end{bmatrix}$$

[*xxx* _]*dbname* _ CSI_PDM_MACS *mac1* [,*mac2…n*]

$$\begin{bmatrix}\textbf{/GROUP}\\ \textbf{/SYSTEM}\\ \textbf{/TABLE = CSI\_PDM\_}\textbf{\textit{pdmname}}\end{bmatrix}$$

Description	*Optional*. Specifies the level of the logical name.	
Options	/GROUP	Groupwide
	/SYSTEM	Systemwide
	/TABLE=CSI_PDM_*pdmname*	User logical name table for a multiple systemwide PDM.

[*xxx*_]*dbname*_CSI_PDM_MACS *mac1* [,*mac2...n*]

Description *Required*. Identifies a list of machines that can be used on the specified database, in order of preference (the preferred machine list).

Format

[*xxx*_]	*Optional*. 1- to 3-character database prefix.
dbname	6-alphanumeric-character database name forming the variable part of the logical name. The first character must be alphabetic.
mac1	Node name of the first choice machine on which the database can be used.
[,*mac2...n*]	Node names of any other machines on which the database can be used in order of preference.

Considerations

- Create a preferred machine list for each database to be used.
- If the preferred machine list contains more than one machine, then duplicate the logical assignments on each machine that might execute an application.
- The logical name *dbname*_CSI_PDM_MACS must be accessible to all applications that might access the specified database.
- You may prefer to make these logical assignments elsewhere, either manually or at system startup. In this case, they are not needed in the PDM initiator script. Because these logical names must exist somewhere on the system, you may prefer to duplicate them in your PDM initiator script rather than to risk omitting them.
- The PDM can load a database on an alternative machine provided the physical files it uses are accessible from that machine (they reside on clustered disks, dual-ported disks, or disks that are local to the active machine).
- The 1- to 3-character prefix allows you to specify a different preferred machine list for databases with the same name in the same PDM.

Understanding which directory the utilities operate in

Each DBA utility executes as a background task, thus freeing the terminal for other uses.

VMS

The utilities operate in your root directory—the directory to which you are set at login.

UNIX

The utilities operate in your current working directory.

If the data sets for the database you are changing are not in your root directory (VMS) or your current working directory (UNIX), you must meet the following conditions:

Environment	Condition
VMS & UNIX	The database description must explicitly state the full file specification for each data set either as a real file specification or as a logical name.
VMS only	The logical assignments for BGRN and CSTUFMT must be available. For a discussion of the LOGICALS.COM procedure, refer to the *SUPRA Server PDM System Administration Guide (VMS)*, P25-0130.

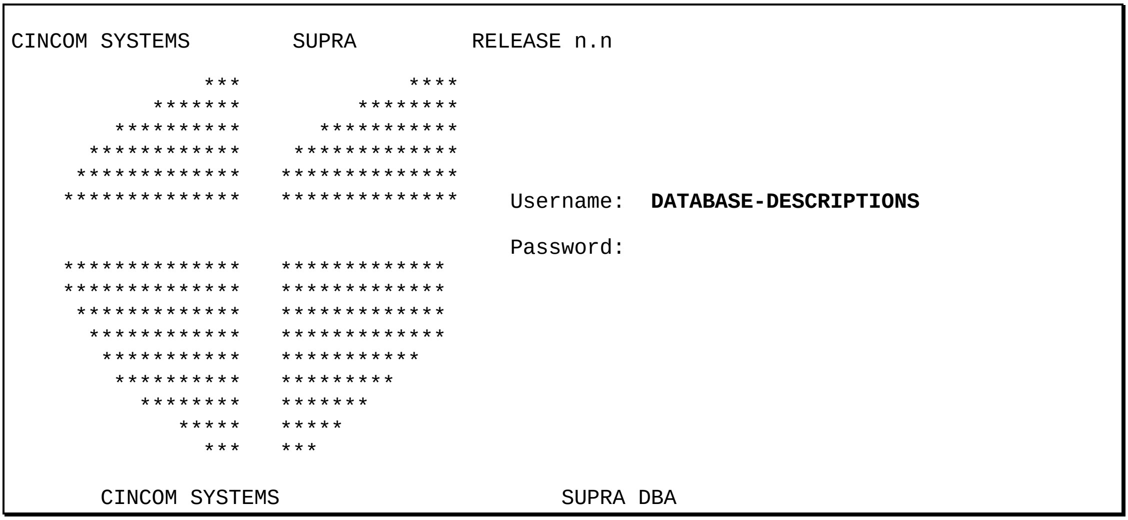
Accessing DBA utilities

To access DBA utilities, first sign on to SUPRA DBA by selecting the DBA Functions option from the SUPRA Facilities screen or by entering the following commands:

VMS `$RUN SUPRA_EXE:CSIDBA   or  $DBA`

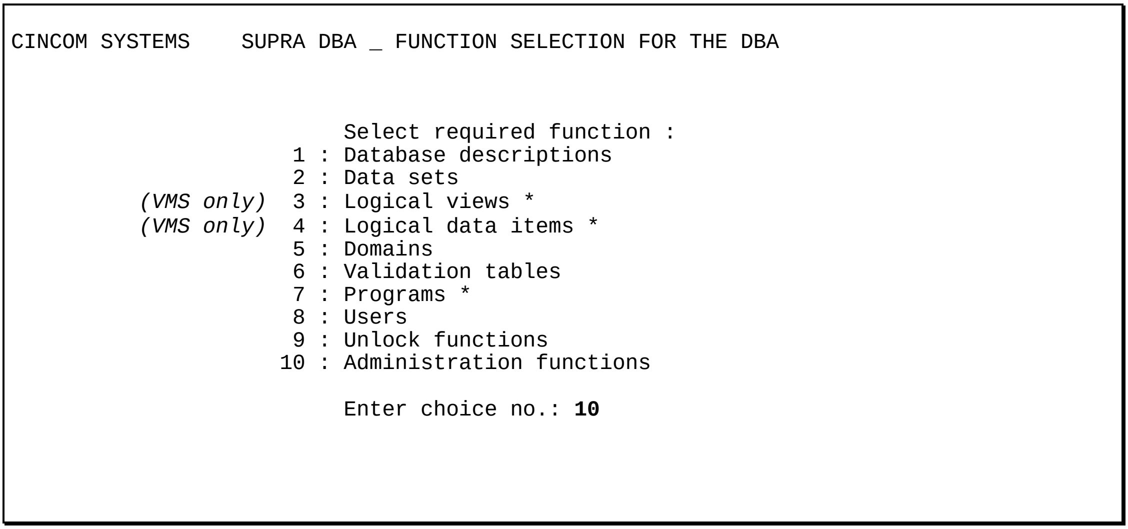
UNIX `SUPRA_EXEC:CSIDBA`

This displays the initial SUPRA sign-on screen similar to the one below. Sign on with your valid user name/password combination.



```
CINCOM SYSTEMS        SUPRA         RELEASE n.n

               ***             ****
           *******         ********
        **********      ***********
      ************    *************
     *************   **************
    **************   **************   Username:  DATABASE-DESCRIPTIONS

                                      Password:
    **************   *************
    **************   *************
     *************   *************
      ************   *************
       ***********   ***********
        **********   *********
          ********   *******
             *****   *****
               ***   ***

       CINCOM SYSTEMS                      SUPRA DBA
```

This displays the Function Selection for the DBA screen; select option 10, Administration functions.

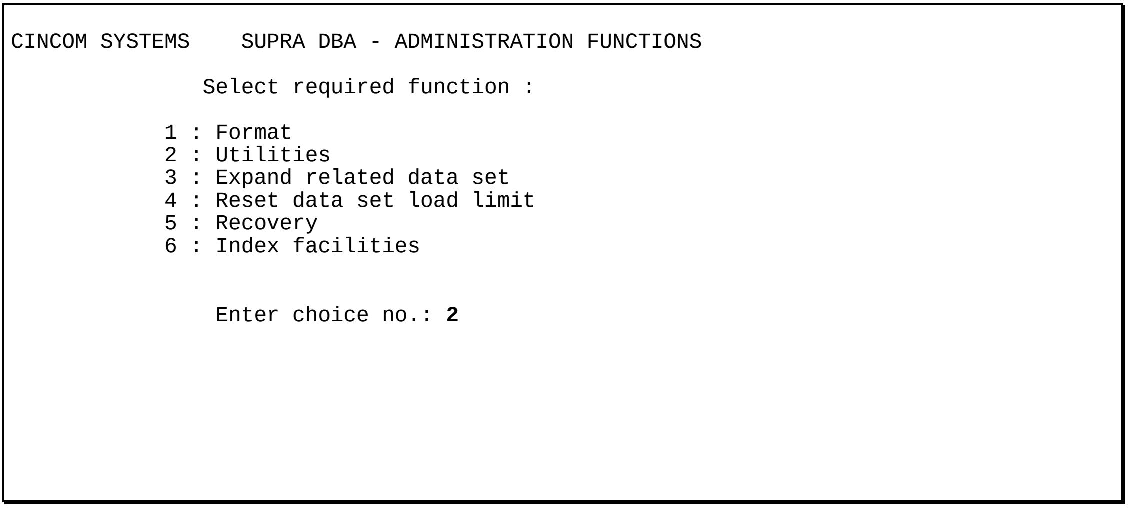


```
CINCOM SYSTEMS    SUPRA DBA _ FUNCTION SELECTION FOR THE DBA



                          Select required function :
                      1 : Database descriptions
                      2 : Data sets
          (VMS only)  3 : Logical views *
          (VMS only)  4 : Logical data items *
                      5 : Domains
                      6 : Validation tables
                      7 : Programs *
                      8 : Users
                      9 : Unlock functions
                     10 : Administration functions

                          Enter choice no.: 10
```

UNIX

* The word *Reserved* displays next to these options.

Select option 2, Utilities, from the Administration Functions menu to begin utility processing.

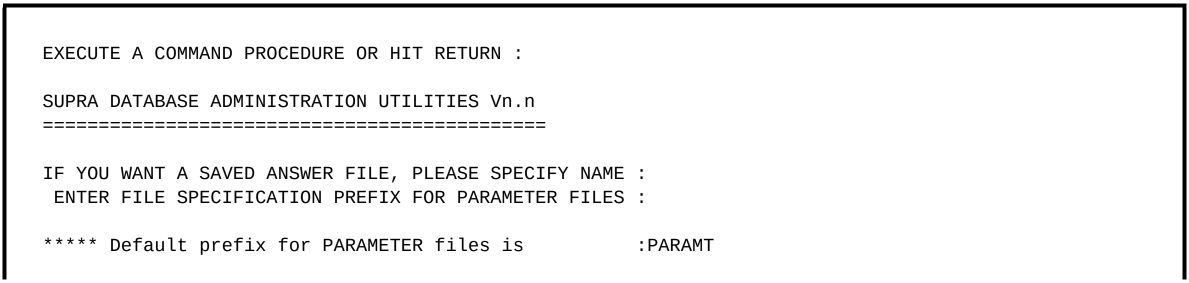


```
CINCOM SYSTEMS    SUPRA DBA - ADMINISTRATION FUNCTIONS

               Select required function :

            1 : Format
            2 : Utilities
            3 : Expand related data set
            4 : Reset data set load limit
            5 : Recovery
            6 : Index facilities


                Enter choice no.: 2
```

Initiating DBA utilities

VMS

When you select Utilities from the DBA Administration Functions menu, DBA prompts you to either execute a command file from a previous utilities run or continue processing with the DBA utilities. Press RETURN and DBA prompts for the name of the saved answer file and the file-specification prefix for the parameter file. To exit from the DBA utilities at any time, press CTRL-Z.



```
EXECUTE A COMMAND PROCEDURE OR HIT RETURN :

SUPRA DATABASE ADMINISTRATION UTILITIES Vn.n
=============================================

IF YOU WANT A SAVED ANSWER FILE, PLEASE SPECIFY NAME :
 ENTER FILE SPECIFICATION PREFIX FOR PARAMETER FILES :

***** Default prefix for PARAMETER files is          :PARAMT
```

UNIX

When you select the Utilities option from the DBA Administration Function menu, DBA prompts you for the path to two input files called UTLNAT.SEQ and UTLERR.MSG. (These files are supplied by Cincom and can be found in the subdirectory bin in the install directory. You can use the path /supra1/$SUPRA1_RELEASE_NUMBER/bin to locate these files.) Utilities will locate the two input files by locating the logical name SUPRA_EXE. This logical name points to a directory containing SUPRA images and the utilities input files. You can define the SUPRA_EXE logical name by using the csideflog command, and making it equal to the path containing UTLNAT.SEQ and UTLERR.MSG. For example, if these two files are found in the path /supra/bin, then prior to running DBA enter the following on the command line:

```
csideflog -s SUPRA_EXE /supra/bin
```

Whether or not you define the path name to the DBA utilities' input files with the SUPRA_EXE logical name, DBA prompts you to either execute a command file from a previous utilities run or continue processing with the DBA utilities. Press Return and DBA prompts for the name of the saved answer file and the specification prefix for the parameter file.

To exit from the utilities at any time press Ctrl-d.

```
ENTER PATH TO UTLNAT.SEQ:
EXECUTE A COMMAND PROCEDURE OR HIT RETURN :

SUPRA DATABASE ADMINISTRATION UTILITIES Vn.n
=============================================

IF YOU WANT A SAVED ANSWER FILE, PLEASE SPECIFY NAME :
 ENTER FILE SPECIFICATION PREFIX FOR PARAMETER FILES :

***** Default prefix for PARAMETER files is          :PARAMT
```

ENTER A PATH TO UTLNAT.SEQ: UNIX

Description *Required*. Displayed only if the environment variable DBA_UTLPATH has not been defined. Enter the path to the UTLNAT.SEQ file to be used as input by the utilities.

Format Valid UNIX pathname excluding the file name UTLNAT.SEQ

Consideration Do not include the UTLNAT.SEQ file name in the path because DBA adds /UTLNAT.SEQ to the specified path to complete a full pathname.

EXECUTE A COMMAND PROCEDURE OR HIT RETURN

Description *Optional*. Enter the file specification of the saved answer file to be processed by the background task.

Format VMS Valid file specification preceded by @

UNIX Valid pathname

Consideration To process a utility, press RETURN without entering a value.

IF YOU WANT A SAVED ANSWER FILE, PLEASE SPECIFY NAME

Description *Optional*. Enter the name of the file to be used as an indirect command file if you execute utilities as a background task. Not required if the background task is to be executed from this session.

Default USAVE.COM in the current directory

Format VMS Valid file specification

UNIX Valid pathname

Consideration Press RETURN to bypass this option.

ENTER FILE SPECIFICATION PREFIX FOR PARAMETER FILES

Description *Required*. Specifies the name of the file or files to contain the control statements for the background task (the background task will later attach an extension to this name).

Default PARAMT

Format 1–9 alphanumeric characters

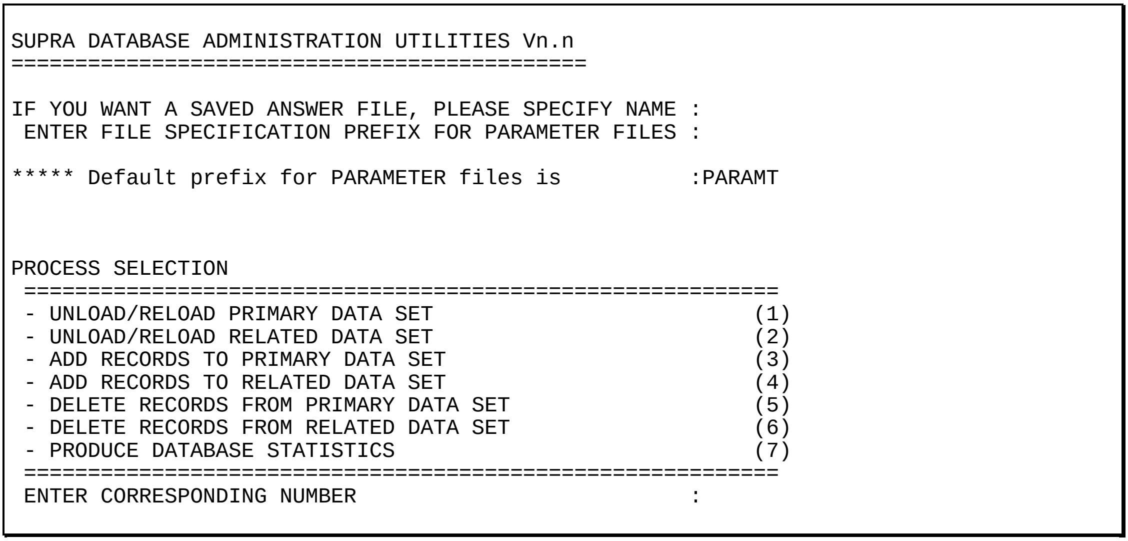
Considerations

- The background task assigns the file extension .PR*n* where *n* depends on the function being performed.

- These files are deleted when the background task completes.

- This prefix is also used for the .LOG file, which contains the results of the utility execution (the number of records in and out, and the error count).

After you respond to the first three prompts, the Process Selection menu appears. Enter the number of the utility you want to process. The remaining prompts depend on the utility you select.



```
SUPRA DATABASE ADMINISTRATION UTILITIES Vn.n
=============================================

IF YOU WANT A SAVED ANSWER FILE, PLEASE SPECIFY NAME :
 ENTER FILE SPECIFICATION PREFIX FOR PARAMETER FILES :

***** Default prefix for PARAMETER files is          :PARAMT



PROCESS SELECTION
 ===========================================================
 - UNLOAD/RELOAD PRIMARY DATA SET                         (1)
 - UNLOAD/RELOAD RELATED DATA SET                         (2)
 - ADD RECORDS TO PRIMARY DATA SET                        (3)
 - ADD RECORDS TO RELATED DATA SET                        (4)
 - DELETE RECORDS FROM PRIMARY DATA SET                   (5)
 - DELETE RECORDS FROM RELATED DATA SET                   (6)
 - PRODUCE DATABASE STATISTICS                            (7)
 ===========================================================
 ENTER CORRESPONDING NUMBER                          :
```

Using the Unload/Reload utility

The Unload/Reload utility allows you to:

- Delete data items from a data set
- Add items to a data set
- Expand the physical characteristics of a data set without losing user-entered records
- Copy data items from one source data set to a target data set in another database
- Repair a corrupted data set by unloading and reloading to the same data set and database description

The following table is an overview of the steps you take to unload and reload data. Each step is described in detail in the specified section.

Step	Explanation	See
1. Rename the source database description file specification from the command line.	Allows you to accept the default when you compile the newly modified database description.	"Renaming the database file specification" on page 50
2. Use SUPRA DBA functions to modify the data set as you require.	Allows you to change the physical file specification of the data set (to accommodate addition, deletion, or expansion of data-items) and rename it as the target data set.	"Modifying the data set's physical file specification" on page 51
3. Validate and compile the modified database description file by accepting the default.	Provides two compiled database description files: the target file just created and the source file you renamed.	"Validating and compiling the new database description" on page 55
4. If the database is new and has a task log or system log defined, format each log.	If a task log and a system log have been defined, they must be formatted.	"Formatting the task and system logs" on page 58

Step	Explanation	See
5. Enter DBA, run Utilities.	Allows you to select the Unload/ Reload utility.	"Unloading and reloading a primary data set" on page 60
6. Respond to the prompts.	Allows you to add data items to, delete data items from, or expand the size of data items.	"Deleting data items from a data set" on page 67 "Adding data items to a data set" on page 69 "Expanding the size of a data item" on page 72
7. Execute the Unload/ Reload utility.	Performs the reload of the data set.	"Executing DBA utilities" on page 75
8. VMS Reset the related linkpath if you used high-speed, secondary linkpath reloading.	Changes the linkpaths back to their original data item names.	"Using high-speed secondary linkpath reloading" on page 79
9. Recompile any other database descriptions connected to the reloaded data sets.	Ensures that the updated data set will be accurate in all connected databases as well.	"Recompiling all connected databases" on page 103
10. Back up the database and format a new system log.		

NOTE

SUPRA DBA cannot unload more than 998 data items per data set file. Therefore, ensure that the data set files you wish to process contain 998 data items or less.

NOTE

If the target database for the Unload/Reload utility is a newly created database, you must define two logical names: one for the compiled database description and one for the preferred machine list. See "Defining a logical name for the compiled database description file" on page 32 and "Defining a logical name for the preferred machine list" on page 36 for procedures to define these logical names.

VMS

The changes you make may affect some views that access the data set. For instance, you may delete a data item, which is accessed by a view. Modify any affected views before you allow application programs to access the database.

To modify both primary data sets and the related data sets to which they are linked, you must go through the steps twice, once to modify the primary data sets and once to modify the related data sets to ensure primary linkpath integrity.

NOTE

Be sure other users cannot sign on to or use this database when you unload or reload.

Renaming the database file specification

Before you invoke utilities, rename the file specification of the compiled database using the VMS RENAME or the UNIX mv command. The database description held in this file is the source database description; it contains the unmodified details of the database and data sets. To change the name of the compiled database description file of a database called UTILDB, enter the following:

VMS
```
$ RENAME UTILDB.MOD UTILDB.SRC
```

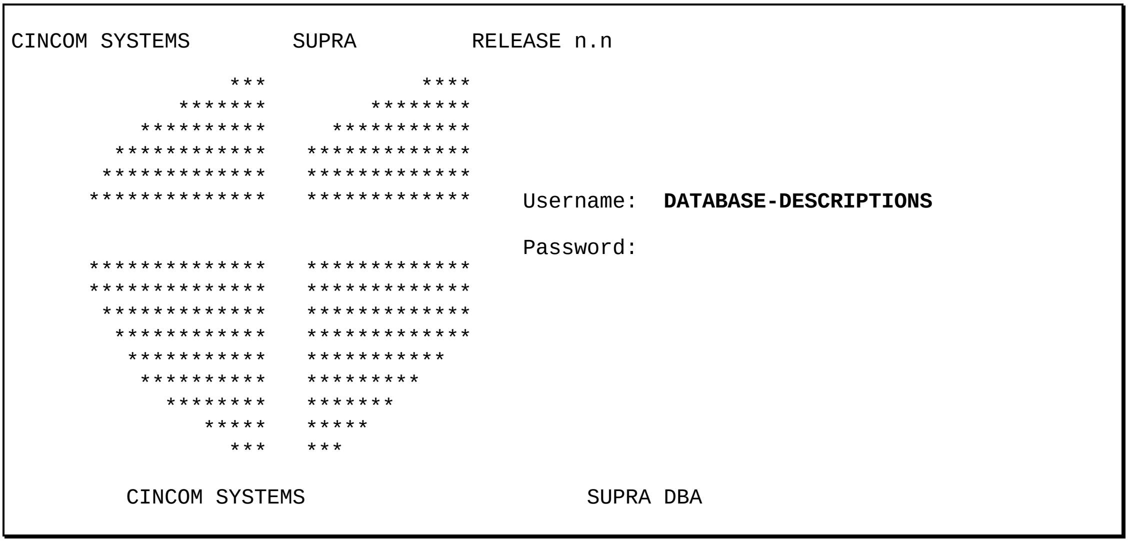
UNIX
```
mv utildb.mod utildb.src
```

When you compile the database after changing the data set to be processed, you can accept the default file specification for the compiled database so the new database description is held in UTILDB.MOD. This database description is the target database description.

You may also define a logical name for the renamed source database description file. See "Defining a logical name for the compiled database description file" on page 32 for the procedures on defining a logical name for the database description file. Once you have defined the logical name, you may enter it at the SOURCE DATABASE DESCRIPTION FILE-SPEC and the TARGET DATABASE DESCRIPTION FILE-SPEC prompts instead of the fully expanded file specifications or pathnames.

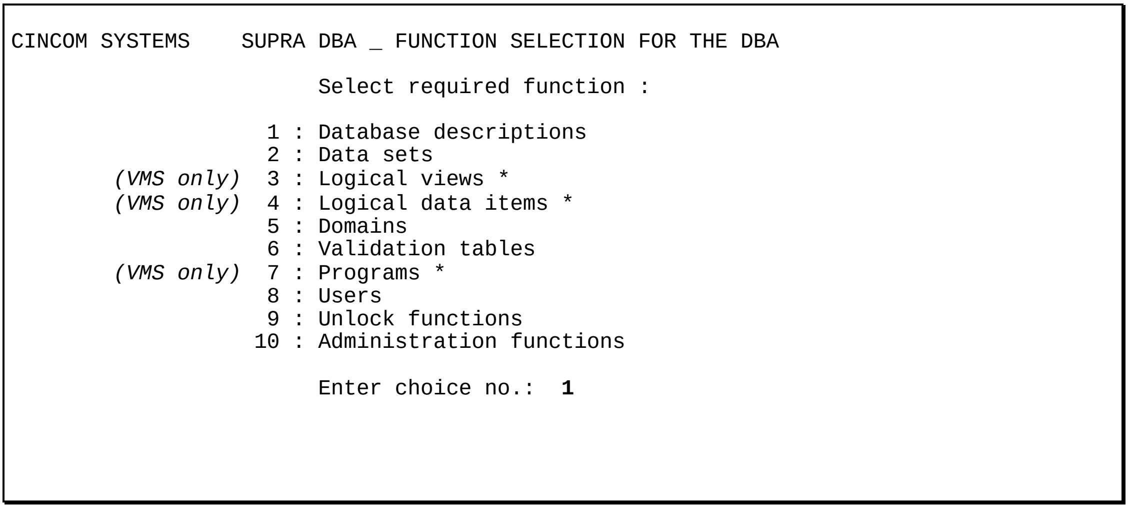
Modifying the data set’s physical file specification

After you rename the source database description file, sign on to DBA.



```
CINCOM SYSTEMS        SUPRA         RELEASE n.n

                 ***            ****
             *******        ********
          **********     ***********
        ************   *************
       *************   *************
      **************   *************    Username:  DATABASE-DESCRIPTIONS

                                        Password:
      **************   *************
      **************   *************
       *************   *************
        ************   *************
         ***********   ***********
          **********   *********
            ********   *******
               *****   *****
                 ***   ***

         CINCOM SYSTEMS                       SUPRA DBA
```

Select option 1, Database descriptions.

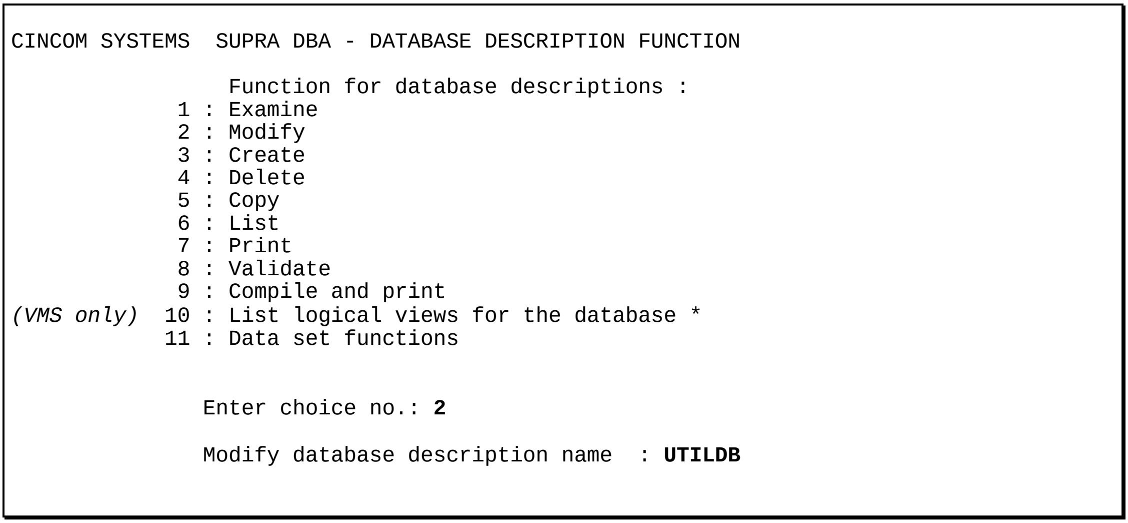


```
CINCOM SYSTEMS    SUPRA DBA _ FUNCTION SELECTION FOR THE DBA

                        Select required function :

                    1 : Database descriptions
                    2 : Data sets
        (VMS only)  3 : Logical views *
        (VMS only)  4 : Logical data items *
                    5 : Domains
                    6 : Validation tables
        (VMS only)  7 : Programs *
                    8 : Users
                    9 : Unlock functions
                   10 : Administration functions

                        Enter choice no.:  1
```

UNIX

* The word *Reserved* displays next to these options.

SUPRA presents the Database Description Function menu. Select option 2, Modify, and enter the name of the database.

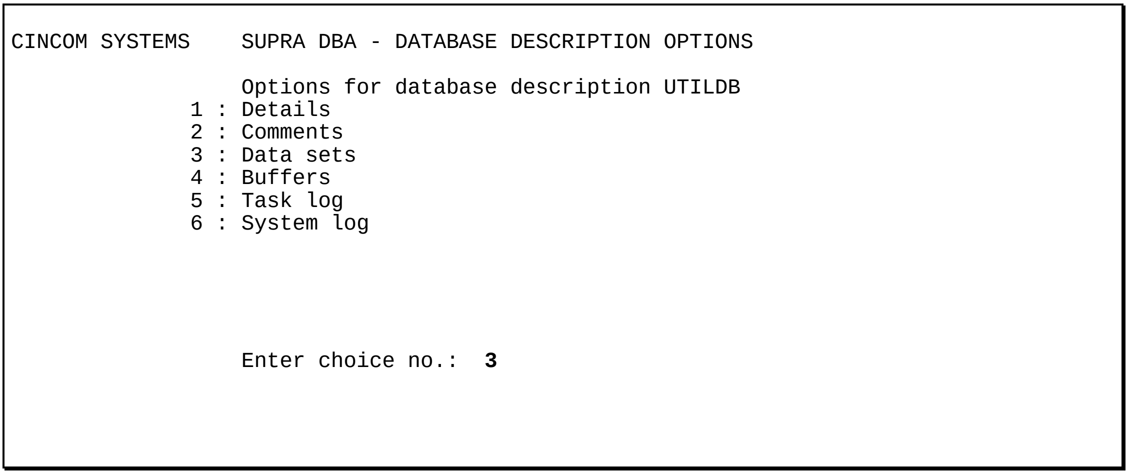


```
CINCOM SYSTEMS  SUPRA DBA - DATABASE DESCRIPTION FUNCTION

                 Function for database descriptions :
             1 : Examine
             2 : Modify
             3 : Create
             4 : Delete
             5 : Copy
             6 : List
             7 : Print
             8 : Validate
             9 : Compile and print
(VMS only)  10 : List logical views for the database *
            11 : Data set functions


               Enter choice no.: 2

               Modify database description name  : UTILDB
```

UNIX

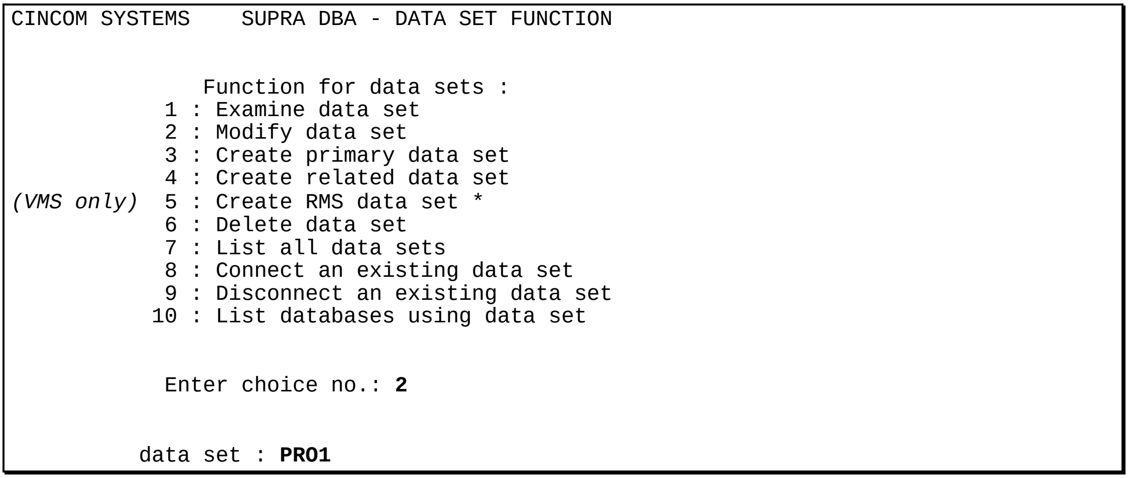
* The word *Reserved* displays next to this option.

The Database Description Options menu displays. Select option 3, Data sets, to display the Data Set Function menu.



```
CINCOM SYSTEMS    SUPRA DBA - DATABASE DESCRIPTION OPTIONS

                  Options for database description UTILDB
              1 : Details
              2 : Comments
              3 : Data sets
              4 : Buffers
              5 : Task log
              6 : System log





                  Enter choice no.:  3
```

Select option 2, Modify data set, and enter the name of the first data set you wish to change, PRO1 in this example.

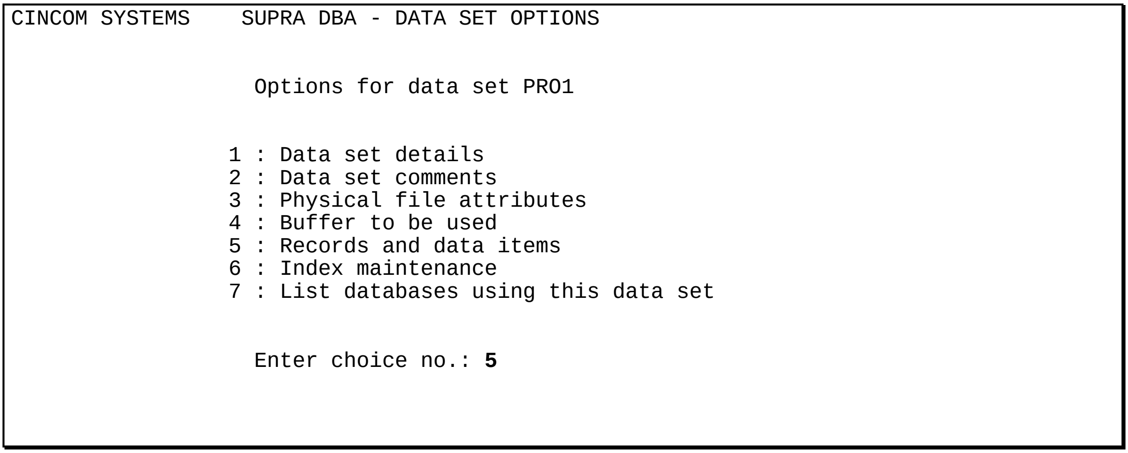


```
CINCOM SYSTEMS    SUPRA DBA - DATA SET FUNCTION


               Function for data sets :
            1 : Examine data set
            2 : Modify data set
            3 : Create primary data set
            4 : Create related data set
(VMS only)  5 : Create RMS data set *
            6 : Delete data set
            7 : List all data sets
            8 : Connect an existing data set
            9 : Disconnect an existing data set
           10 : List databases using data set


            Enter choice no.: 2


          data set : PRO1
```

UNIX

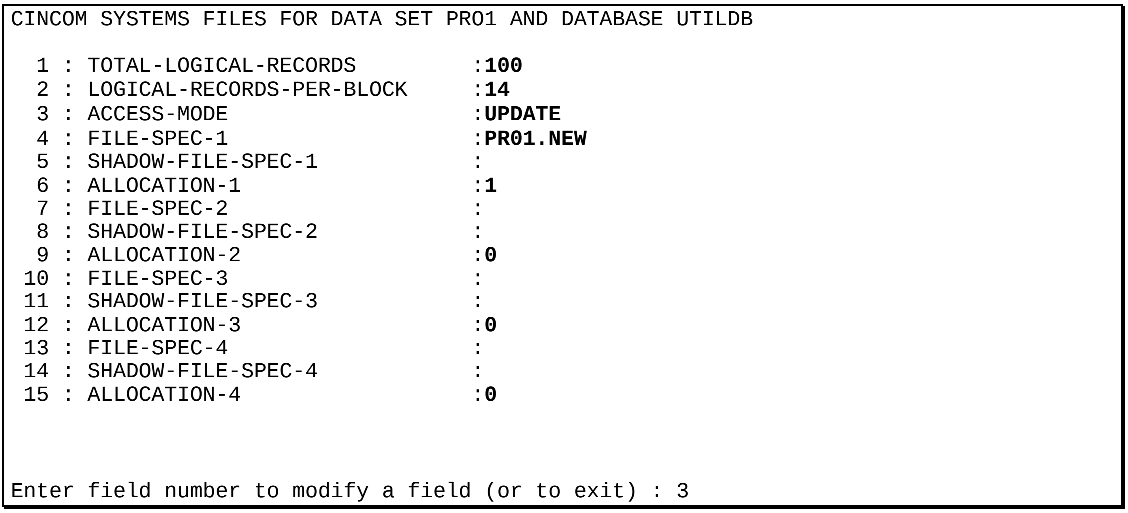
* The word *Reserved* displays next to this option.

Once you specify the name of the data set you wish to modify, DBA displays the Data Set Options screen. Select option 5, Records and data items, to change the record layout of the data set.



```
CINCOM SYSTEMS    SUPRA DBA - DATA SET OPTIONS


                   Options for data set PRO1


                 1 : Data set details
                 2 : Data set comments
                 3 : Physical file attributes
                 4 : Buffer to be used
                 5 : Records and data items
                 6 : Index maintenance
                 7 : List databases using this data set


                   Enter choice no.: 5
```

You must change the file specification for each modified target data set so it is on a different physical file. For example, change the file specification for the target data set PRO1 to PRO1.NEW as shown below:



```
CINCOM SYSTEMS FILES FOR DATA SET PRO1 AND DATABASE UTILDB

  1 : TOTAL-LOGICAL-RECORDS         :100
  2 : LOGICAL-RECORDS-PER-BLOCK     :14
  3 : ACCESS-MODE                   :UPDATE
  4 : FILE-SPEC-1                   :PR01.NEW
  5 : SHADOW-FILE-SPEC-1            :
  6 : ALLOCATION-1                  :1
  7 : FILE-SPEC-2                   :
  8 : SHADOW-FILE-SPEC-2            :
  9 : ALLOCATION-2                  :0
 10 : FILE-SPEC-3                   :
 11 : SHADOW-FILE-SPEC-3            :
 12 : ALLOCATION-3                  :0
 13 : FILE-SPEC-4                   :
 14 : SHADOW-FILE-SPEC-4            :
 15 : ALLOCATION-4                  :0



Enter field number to modify a field (or to exit) : 3
```

If you are changing only primary data sets or only related data sets, modify the physical file specification of all the data sets to be changed. If you wish to change both primary data sets and their connected related data sets, you must modify the database in two stages:

1. Modify the physical file specification of all the primary data sets and process them using the Unload/Reload utility.

2. Modify the physical file specification of all the related data sets and process them using the Unload/Reload utility.

Unmodified data sets are still present and unchanged in the target database description.

Warning: If you attempt to modify both primary and connected related data sets in a single run, you will corrupt the linkpaths.

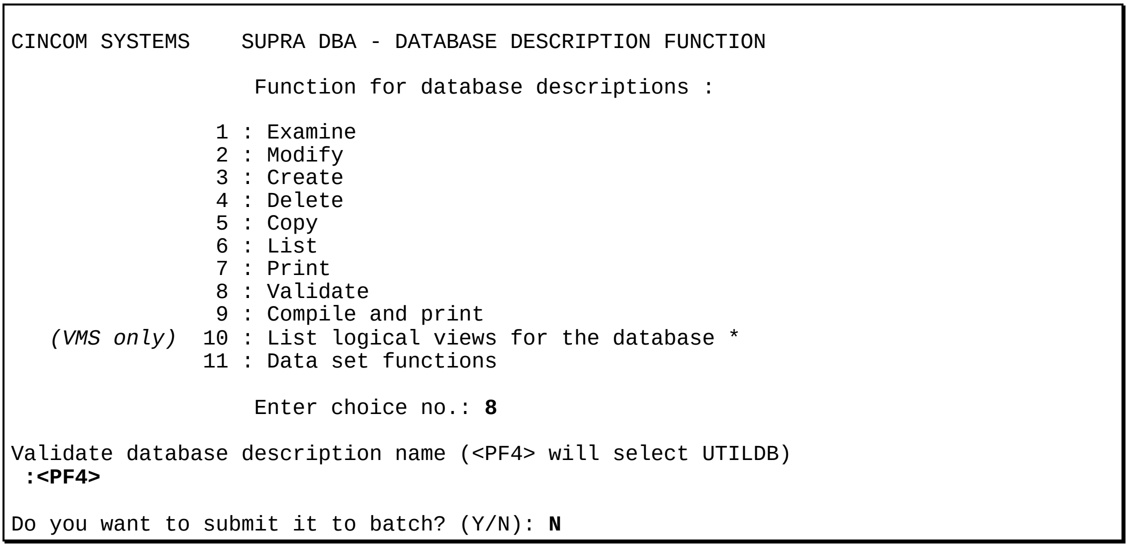
Validating and compiling the new database description

After you make any data set modifications and specify a new file specification for each data set to be unloaded and reloaded, validate the database description. Press PF1 as many times as necessary to return to the Database Description Function menu; then select option 8, Validate. DBA displays a message indicating success or failure when it completes validation.

- If unsuccessful, correct the errors and revalidate
- If successful, compile the new database description by answering Y to the “Do you want to compile it (Y, N) :” prompt

NOTE

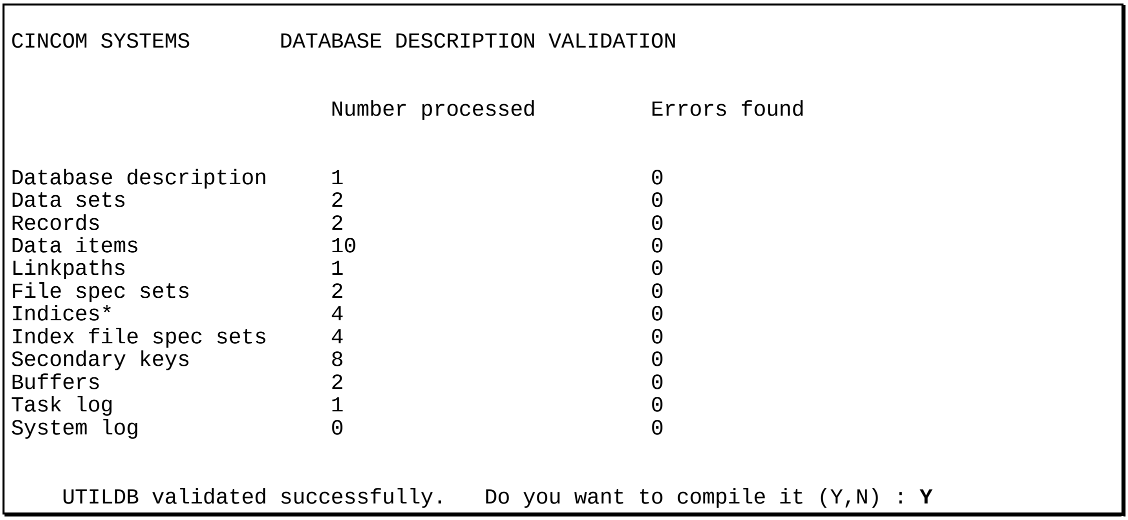
You can also use Batch Validate and Compile to create the target database description.



```
CINCOM SYSTEMS    SUPRA DBA - DATABASE DESCRIPTION FUNCTION

                   Function for database descriptions :

                1 : Examine
                2 : Modify
                3 : Create
                4 : Delete
                5 : Copy
                6 : List
                7 : Print
                8 : Validate
                9 : Compile and print
   (VMS only)  10 : List logical views for the database *
               11 : Data set functions

                   Enter choice no.: 8

Validate database description name (<PF4> will select UTILDB)
 :<PF4>

Do you want to submit it to batch? (Y/N): N
```

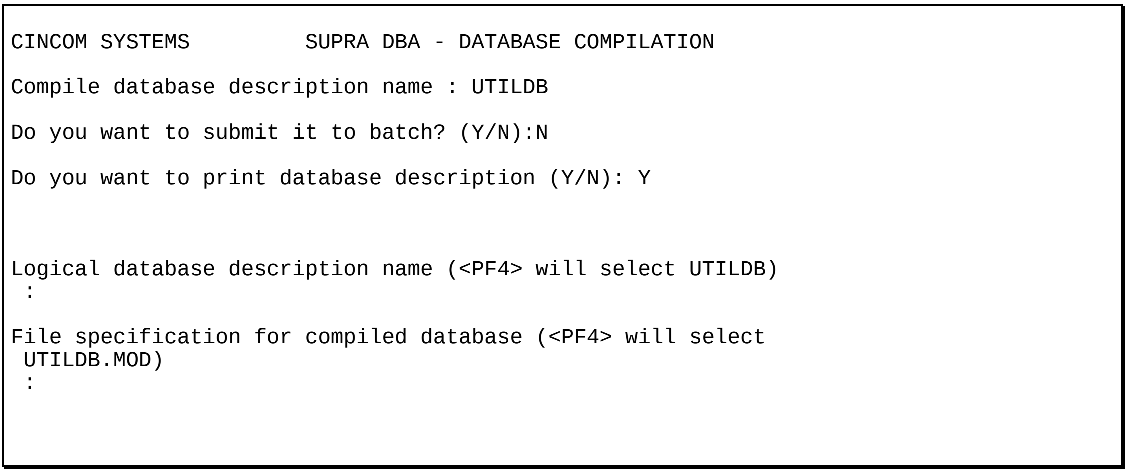
UNIX

* The word *Reserved* displays next to this option.



```
CINCOM SYSTEMS       DATABASE DESCRIPTION VALIDATION


                         Number processed         Errors found


Database description     1                        0
Data sets                2                        0
Records                  2                        0
Data items               10                       0
Linkpaths                1                        0
File spec sets           2                        0
Indices*                 4                        0
Index file spec sets     4                        0
Secondary keys           8                        0
Buffers                  2                        0
Task log                 1                        0
System log               0                        0


    UTILDB validated successfully.   Do you want to compile it (Y,N) : Y
```

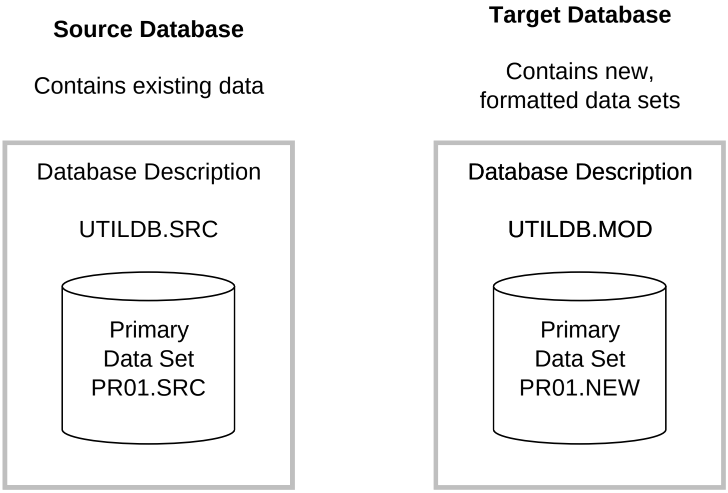
The Database Compilation screen displays next. Change the defaults as appropriate. For example, if you want your application programs to sign on with a different name, change the logical database name.



```
CINCOM SYSTEMS         SUPRA DBA - DATABASE COMPILATION

Compile database description name : UTILDB

Do you want to submit it to batch? (Y/N):N

Do you want to print database description (Y/N): Y



Logical database description name (<PF4> will select UTILDB)
 :

File specification for compiled database (<PF4> will select
 UTILDB.MOD)
 :
```

Because you renamed the source database's description file from UTILDB.MOD to UTILDB.SRC, you can accept the default filename UTILDB.MOD for the compiled target database.

At this point, both the source-database description and the target-database description exist. The following figure shows the result of the previous sequence of steps. The source database has the file specification, UTILDB.SRC, and the target database has the file specification, UTILDB.MOD. Both databases have a primary data set, PR01, but each database has a different physical file specification for the data sets. The data is still in the source database.

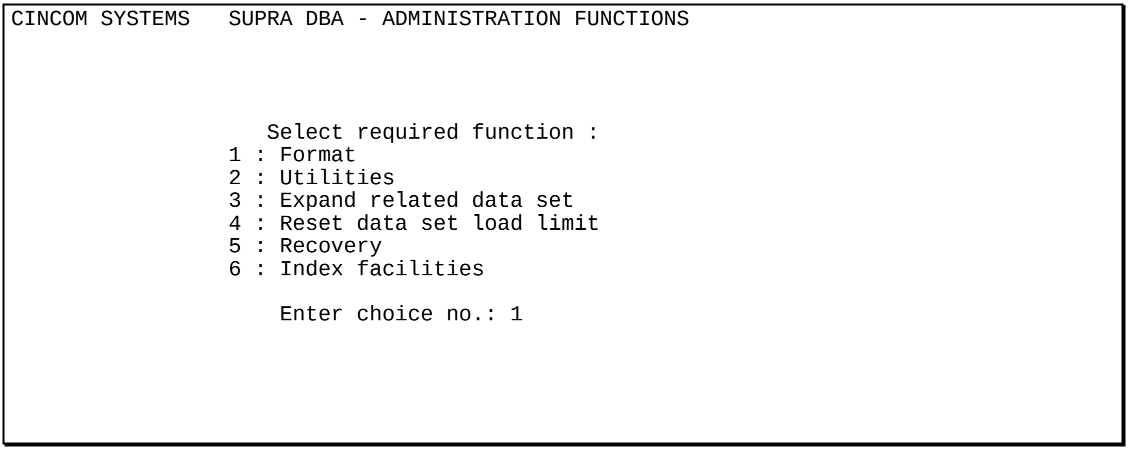


Formatting the task and system logs

If the target database for the Unload/Reload utility is a newly created database, and a task log or system log was defined for it, you must format each log with the DBA Format function. If the target database is an existing database, you may skip this step.

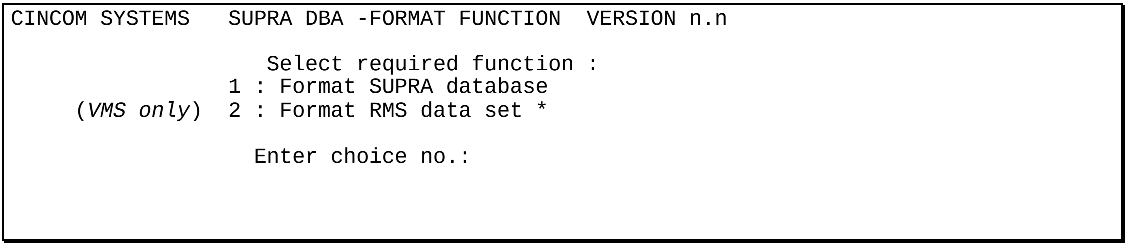
To format the task log, sign on to DBA and select option 10, Administration functions, from the DBA Function Selection menu.

The Administration Functions menu displays. Select option 1, Format.



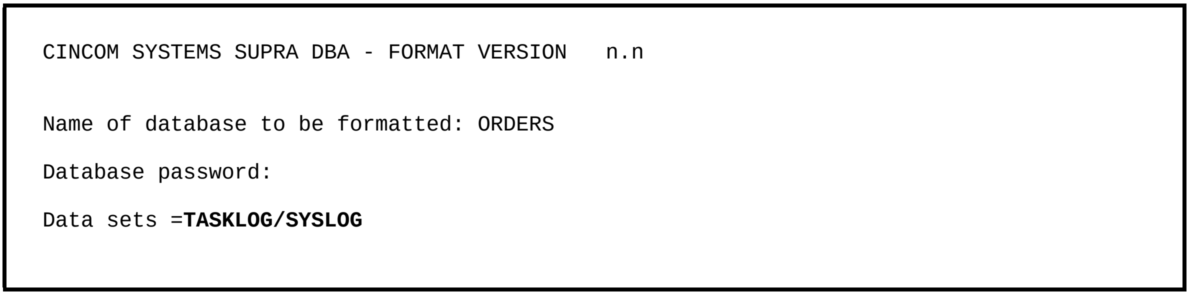
```
CINCOM SYSTEMS   SUPRA DBA - ADMINISTRATION FUNCTIONS




                    Select required function :
                 1 : Format
                 2 : Utilities
                 3 : Expand related data set
                 4 : Reset data set load limit
                 5 : Recovery
                 6 : Index facilities

                     Enter choice no.: 1
```

This displays the Format Function menu. Select option 1, Format SUPRA database.



```
CINCOM SYSTEMS   SUPRA DBA -FORMAT FUNCTION  VERSION n.n

                    Select required function :
                 1 : Format SUPRA database
     (VMS only)  2 : Format RMS data set *

                   Enter choice no.:
```

This displays a screen that prompts for information concerning the database and data sets you want to format, as shown below. Enter the database name and the database password. Use the information following the screen to answer the “Data sets=” prompt.



```
CINCOM SYSTEMS SUPRA DBA - FORMAT VERSION   n.n


Name of database to be formatted: ORDERS

Database password:

Data sets =TASKLOG/SYSLOG
```

data sets =

Description *Required*. Specifies the data sets, task log, and/or system log to be formatted.

When DBA finishes formatting the log, it displays a completion message.

Unloading and reloading a primary data set

VMS

You can use the same procedures to unload and reload both primary and related data sets. However, to accelerate utility processing for related data sets, you may use a different procedure called high-speed secondary linkpath unloading and reloading (see “Using high-speed secondary linkpath reloading” on page 79).

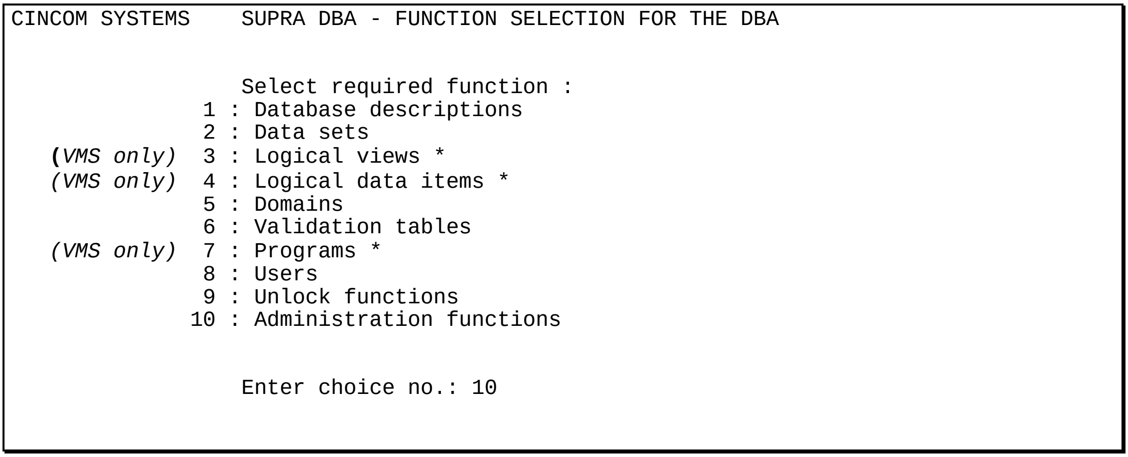
UNIX

The procedures in this section apply to primary data sets only. The procedures you use to unload and reload related data sets vary from primary data set processing and are supplied in “Unloading and reloading a related data set under UNIX” on page 94. To accelerate utility processing for related data sets, use the Fast utilities described in “Using Fast utilities” on page 127.

NOTE

If you use system logging on a reloaded database, ensure that the reload has completed successfully; then back up the database files and format a new system log. Do not allow any users to access the database description until you format a new, system log file.

This section uses the primary data set PR01 to show how to unload and reload data items. To initiate Utilities select option 10, Administration functions, from the Function Selection for the DBA menu:

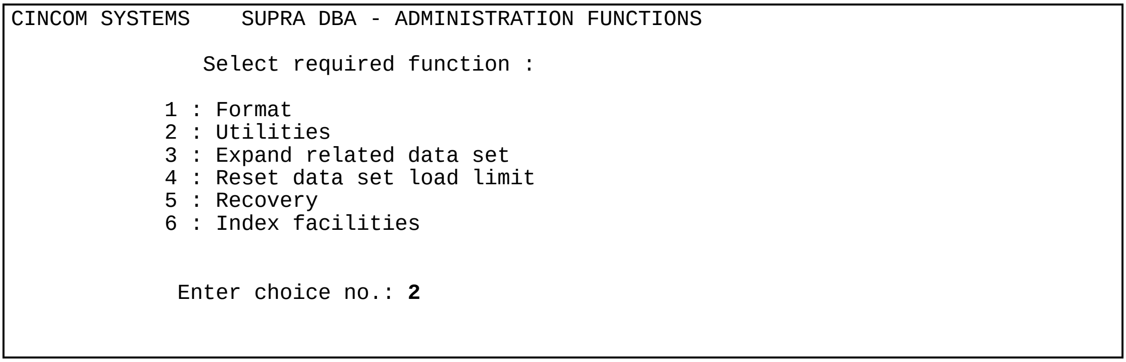


```
CINCOM SYSTEMS    SUPRA DBA - FUNCTION SELECTION FOR THE DBA


                  Select required function :
               1 : Database descriptions
               2 : Data sets
   (VMS only)  3 : Logical views *
   (VMS only)  4 : Logical data items *
               5 : Domains
               6 : Validation tables
   (VMS only)  7 : Programs *
               8 : Users
               9 : Unlock functions
              10 : Administration functions


                  Enter choice no.: 10
```

UNIX

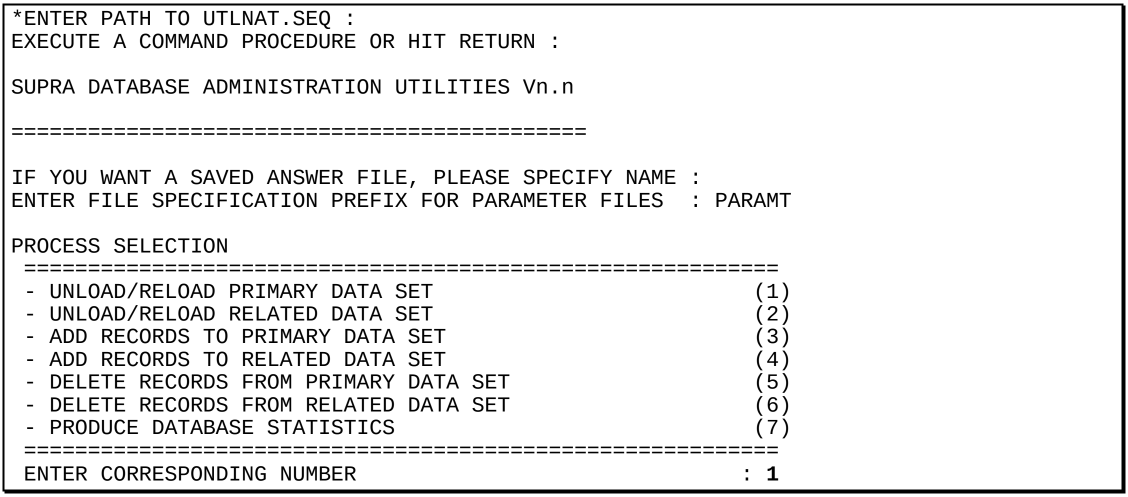
* The word *Reserved* displays next to these options.

Select option 2, Utilities, from the Administration Functions menu :



```
CINCOM SYSTEMS    SUPRA DBA - ADMINISTRATION FUNCTIONS

               Select required function :

            1 : Format
            2 : Utilities
            3 : Expand related data set
            4 : Reset data set load limit
            5 : Recovery
            6 : Index facilities


             Enter choice no.: 2
```

DBA presents the Process Selection menu. Specify a name for the saved answer file if you wish to keep the answers from this session. Specify the file specification prefix for the parameter files unless you want to use the default, PARAMT. Select option 1, UNLOAD/RELOAD PRIMARY DATA SET (if you are using these procedures for a related data set, select option 2).



```
*ENTER PATH TO UTLNAT.SEQ :
EXECUTE A COMMAND PROCEDURE OR HIT RETURN :

SUPRA DATABASE ADMINISTRATION UTILITIES Vn.n

=============================================

IF YOU WANT A SAVED ANSWER FILE, PLEASE SPECIFY NAME :
ENTER FILE SPECIFICATION PREFIX FOR PARAMETER FILES  : PARAMT

PROCESS SELECTION
 ===========================================================
 - UNLOAD/RELOAD PRIMARY DATA SET                         (1)
 - UNLOAD/RELOAD RELATED DATA SET                         (2)
 - ADD RECORDS TO PRIMARY DATA SET                        (3)
 - ADD RECORDS TO RELATED DATA SET                        (4)
 - DELETE RECORDS FROM PRIMARY DATA SET                   (5)
 - DELETE RECORDS FROM RELATED DATA SET                   (6)
 - PRODUCE DATABASE STATISTICS                            (7)
 ===========================================================
 ENTER CORRESPONDING NUMBER                              : 1
```

UNIX

* UNIX only

After you select this function, DBA prompts you for the source and target database identification. Default values and some sample user input are included for illustration. (User input is lowercase under UNIX.)

```
SOURCE DATABASE DESCRIPTION FILE-SPEC         : UTILDB.SRC
SOURCE DATABASE NAME                          : UTILDB
SOURCE DATA SET NAME                          : PR01.
SOURCE DATABASE PASSWORD                      :
TARGET DATABASE DESCRIPTION FILE-SPEC         : UTILDB.MOD
TARGET DATABASE NAME                          : UTILDB
TARGET DATA SET NAME                          : PR01
TARGET DATABASE PASSWORD                      :


WORK FILE SPECIFICATION                       :


*****Default work file specification is        :WORKOUT.TMP


 TAPE LABEL*                                   :
```

VMS

* VMS only

SOURCE DATABASE DESCRIPTION FILE-SPEC

Description *Required*. Specifies the file specification (VMS) or pathname (UNIX) of the source database description.

Format VMS Valid file specification

UNIX Valid pathname

Considerations

- This file specification must match the source file specification used with the VMS $RENAME command or the UNIX mv command.
- VMS If you omit dev:[directory], it defaults to the current directory.

  UNIX If you omit the directory, it defaults to the current directory.
- VMS If you omit the file type, it defaults to .MOD.

  UNIX If you omit the file suffix, it defaults to .mod.
- If you have defined a logical name that points to the source database's description file specification, you may enter the logical name.

SOURCE DATABASE NAME

Description *Required*. Specifies the logical name of the database containing the data set you want to unload.

Format 6 alphanumeric characters

SOURCE DATA SET NAME

Description *Required*. Specifies the primary data set to be unloaded.

Format 4 alphanumeric characters

Consideration This data set must be defined in the source database description and may contain up to 998 data items.

SOURCE DATABASE PASSWORD

Description *Required* if the database description uses a password. This name must match that password.

Format 6 alphanumeric characters

Considerations

- The password does not display.
- If you omit the correct password, DBA displays an error message and reprompts for the password.
- If the database description did not use a password, press RETURN.
- If the password is less than 6 characters, enter spaces to fill 6 characters.

TARGET DATABASE DESCRIPTION FILE-SPEC

Description *Required*. Specifies the file specification of the target database.

Format VMS Valid file specification

UNIX Valid pathname

Considerations

- This file specification must match that of the new database description.
- VMS If you omit dev:[directory], it defaults to the current directory.

  UNIX If you omit the directory, it defaults to the current directory.
- VMS If you omit the file type, it defaults to .MOD.

  UNIX If you omit the file suffix, it defaults to .mod.
- If you have defined a logical name that points to the target database description file, you may enter the logical name.

TARGET DATABASE NAME

Description *Required*. Specifies the logical name of the database containing the data set to be loaded.

Format 6 alphanumeric characters

TARGET DATA SET NAME

Description *Required*. Specifies the primary data set to be loaded.

Format 4 alphanumeric characters

Consideration This data set must be defined in the target database description.

TARGET DATABASE PASSWORD

Description *Required* if the database description uses a password. This name must match that password

Format 6 alphanumeric characters

Considerations

- The password does not display.
- If you omit the correct password, DBA displays an error message and reprompts for the password.
- If the database description did not use a password, press RETURN.
- If the password is less than 6 characters, enter spaces to fill 6 characters.

WORK FILE SPECIFICATION

Description *Optional*. Specifies the file specification of the work file used to hold the data from the unloaded data set and spaces for new data items.

Default WORKOUT.TMP

Format **VMS** Valid file specification

UNIX Valid pathname or tape device name such as /dev/rmt/0m

***** Default work file specification is

Restriction Displayed only if you do not enter a value for the temporary work file specification.

Description Displays the default value for the temporary work file specification, WORKOUT.TMP

TAPE LABEL VMS

Restriction Displayed only if the work file specification indicates a tape drive.

Description *Optional*. Specifies the label of the tape to contain the work file.

Default UTLTAP

Format 6 alphanumeric characters

After you identify the source and target databases and the work file, the DBA utilities issue a prompt for the data items to be unloaded from the source data set into the work file, and then for the data items to be loaded from the work file into the target data set. How you respond to these prompts determines whether data items will be added, deleted, or expanded. See "Deleting data items from a data set" below for information about how to respond to the prompts to delete items from a data set. See "Adding data items to a data set" on page 69 for information about how to respond to the prompts to add items to a data set, and "Expanding the size of a data item" on page 72 for instructions how to respond to the prompts to expand the size of data items in a data set. (The data item lists must correspond to the work file records.)

Deleting data items from a data set

To delete a data item from a data set, omit the data item from the target data set using DBA functions. When deleting items, keep in mind that the minimum record length for a primary data set is 21; the minimum record length for a related data set is 41. Any data held in data items not specified in the target data set is lost. To delete a data item and any data it holds, answer N to the “DO YOU WANT TO COPY ALL SOURCE DATA ITEMS?” prompt and specify only those data items that you wish to unload from the source data set. The Unload/Reload utility copies the specified records from the source data set into a work file.

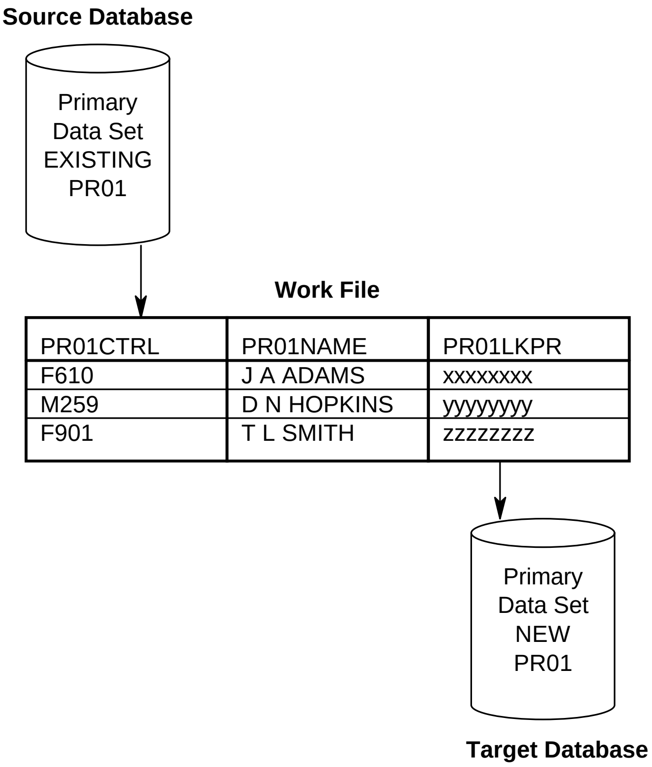
Because the work file records exactly match the data items specified in the target data set, answer Y to the “DO YOU WANT TO COPY ALL TARGET DATA ITEMS:” prompt.

The following screen illustration shows how to delete the data items PR01BRTH and PR01RLSE from the primary data set PR01:

Old data items	New data items
PR01CTRL=4	PR01CTRL=4
PR01NAME=30	PRO1NAME=30
PR01LKPR=8	PR01LKPR=8
PR01BRTH=5	
PR01RLSE=10	

```
DO YOU WANT TO COPY ALL SOURCE DATA ITEMS          ? N
                          **** Data item list ****
PR01CTRL     PR01NAME      PR01LKPR      PR01BRTH      PR01RLSE
END.

REQUIRED SOURCE DATA ITEM NAME                     : PR01CTRL
REQUIRED SOURCE DATA ITEM NAME                     : PR01NAME
REQUIRED SOURCE DATA ITEM NAME                     : PR01LKPR
REQUIRED SOURCE DATA ITEM NAME                     : END.

DO YOU WANT TO COPY ALL TARGET DATA ITEMS          ? Y

***** Target data item is                          : PR01CTRL
***** Target data item is                          : PR01NAME
***** Target data item is                          : PR01LKPR
***** Target data item is                          : END.

>>> Working ...<<<
```

The following figure shows this sequence of events. The DBA utilities unload only the specified data items from the existing PR01 file into the work file, ignoring all data for PR01BRTH and PR01RLSE. The unloaded records are then reloaded into the new PR01 file; the data items PR01BRTH and PR01RLSE have been deleted.



PR01CTRL	PR01NAME	PR01LKPR
F610	J A ADAMS	xxxxxxxx
M259	D N HOPKINS	yyyyyyyy
F901	T L SMITH	zzzzzzzz

NOTE

You can add, delete, and expand data items in one data set in a single unload/reload operation.

Adding data items to a data set

To add a data item to a data set, specify the extra data item only in the target data set. The source data set remains unchanged; therefore, answer Y to the "DO YOU WANT TO COPY ALL SOURCE DATA ITEMS?" prompt to unload all the source records.

VMS When adding items, keep in mind that the maximum record length for any data set is 4096.

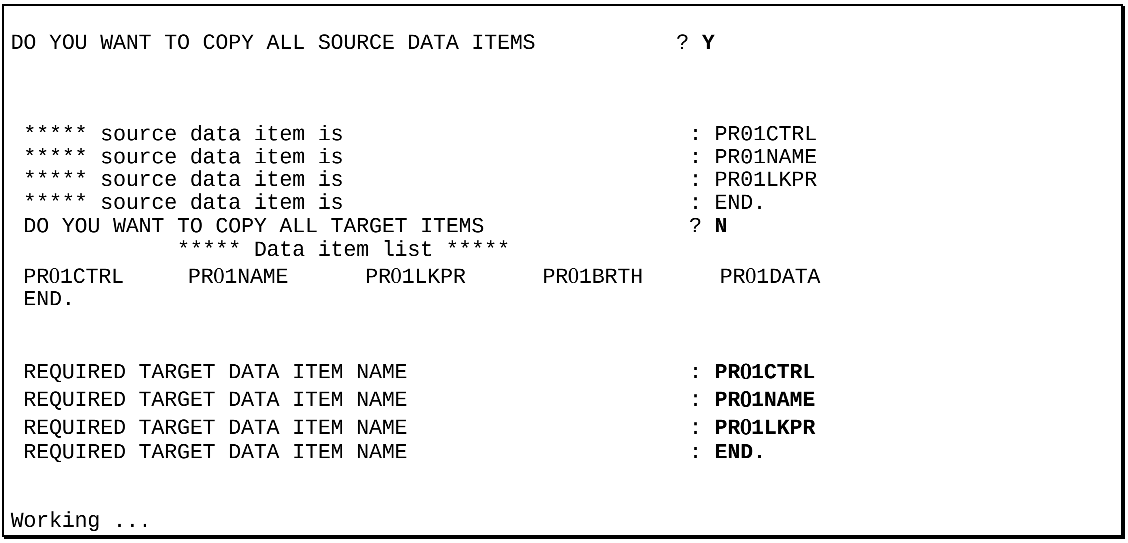
UNIX When adding items, keep in mind that the maximum record length for any data set is 32,768.

The work file records must match the data items specified in the target data set. Because there are no records in the work file to be loaded into the extra target data item, answer N to the "DO YOU WANT TO COPY ALL TARGET DATA ITEMS?" prompt.

Specify only those data items whose records were unloaded from the source data set into the work file. The Unload/Reload utility copies the records from the work file into the target data set and sets the new data item to binary zeros whether it is numeric or character type. If you desire spaces instead of binary zeros, consider using the *FILL technique explained in "Expanding the size of a data item" on page 72.

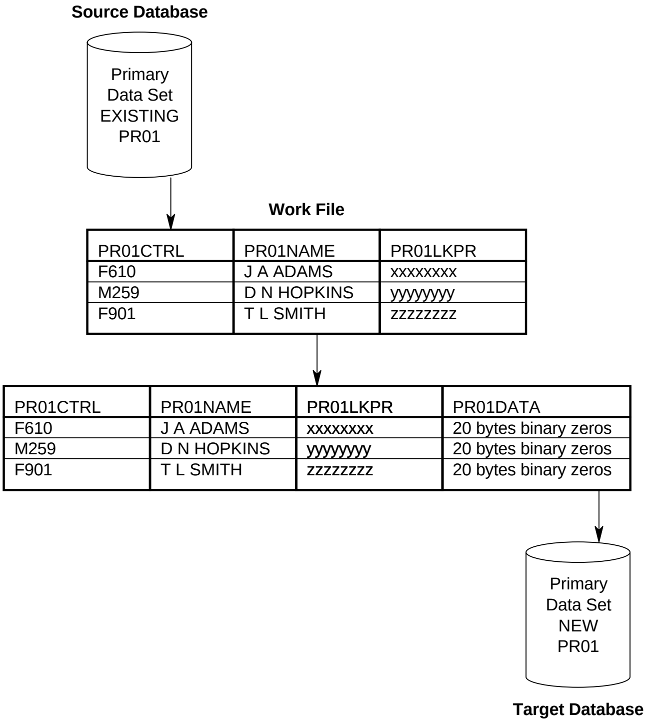
The following screen illustration shows how to add the 20-character data item named PR01DATA to the primary data set PR01:

Old data items	New data items
PR01CTRL=4	PR01CTRL=4
PR01NAME=30	PR01NAME=30
PR01LKPR=8	PR01LKPR=8
	PR01DATA=20



```
DO YOU WANT TO COPY ALL SOURCE DATA ITEMS           ? Y


 ***** source data item is                            : PR01CTRL
 ***** source data item is                            : PR01NAME
 ***** source data item is                            : PR01LKPR
 ***** source data item is                            : END.
 DO YOU WANT TO COPY ALL TARGET ITEMS                 ? N
             ***** Data item list *****
 PR01CTRL     PR01NAME      PR01LKPR      PR01BRTH      PR01DATA
 END.


 REQUIRED TARGET DATA ITEM NAME                       : PR01CTRL
 REQUIRED TARGET DATA ITEM NAME                       : PR01NAME
 REQUIRED TARGET DATA ITEM NAME                       : PR01LKPR
 REQUIRED TARGET DATA ITEM NAME                       : END.


Working ...
```

The following figure shows this sequence of events.



PR01CTRL	PR01NAME	PR01LKPR
F610	J A ADAMS	xxxxxxxx
M259	D N HOPKINS	yyyyyyyy
F901	T L SMITH	zzzzzzzz

PR01CTRL	PR01NAME	PR01LKPR	PR01DATA
F610	J A ADAMS	xxxxxxxx	20 bytes binary zeros
M259	D N HOPKINS	yyyyyyyy	20 bytes binary zeros
F901	T L SMITH	zzzzzzzz	20 bytes binary zeros

The data items are unloaded from the existing PR01 file into the work file. They are then reloaded into the new PR01 file. The additional data item PR01DATA, specified in the new PR01 file, is set to binary zeros.

You can add, delete, and expand data items in one data set in a single unload/reload operation.

To set a data item to something other than binary zeros or spaces, you need to write an RDM application (VMS only) or use PDML commands to go through the reloaded data set, extract the data item, and fill it with whatever data you require.

Expanding the size of a data item

To increase the size of a data item, you must define the new length in the target database description using DBA functions. When you run the Unload/Reload utility, the records unloaded into the work file from the source data set will not be long enough for the target data set.

1. Expand the length of the source record by answering N to the "DO YOU WANT TO COPY ALL SOURCE DATA ITEMS?" prompt.

2. Add the filler data item *FILL=*nn* before the data item to be expanded if you want to add the space to the beginning of the data item. Add *FILL=*nn* after the data item to be expanded if you want to add space to the end of the data item. The value of *nn* is the difference between the source data item length and the target data item length.

3. Answer Y to the "DO YOU WANT TO COPY ALL TARGET DATA ITEMS?" prompt. You will have already modified the target data set, so the data items are the correct length. The Unload/Reload utility then copies the specified items from the source data set into the work file, with additional space in each record to match the *FILL value. This format now corresponds to the target data item descriptions. You can also use *FILL=*nn* to expand data items in coded records. However, when you do so, ensure you follow these rules:

    - Explicitly unload all data items, including those in record codes you do not wish to change. DBA utilities perform a code-directed read and will, therefore, lose data held in coded records that are not specified on the unload.

    - Do not specify *FILL=nn in front of the first coded data item. If you do, the DBA utilities will attempt to expand the base portion of the record.

    - Do not modify linkpaths in the coded portion.

The following screen illustration shows how to add 10 bytes to the beginning of the data item PR01NAME (PR01NAME is in the primary data set PR01CTRL). The length of the source data item PR01NAME plus the *FILL data item is equal to the length of the PR01NAME data item in the target data set.

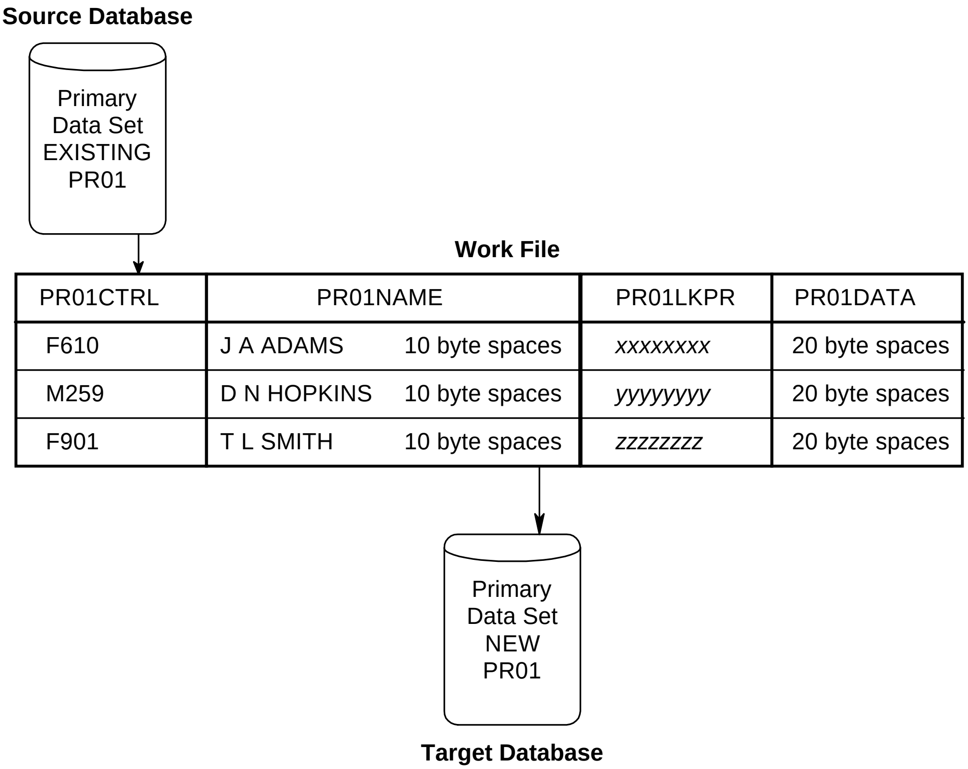
Old data items	New data items
PR01CTRL=4	PR01CTRL=4
PR01NAME=30	PR01NAME=40
PR01LKPR=8	PR01LKPR=8
PR01DATA=20	PR01DATA=20
.	
.	

```
DO YOU WANT TO COPY ALL SOURCE DATA ITEMS          ? N
       ***** Data item list *****
PR01CTRL     PR01NAME      PR01LKPR     PR01DATA            END.


REQUIRED SOURCE DATA ITEM NAME                     : PR01CTRL
REQUIRED SOURCE DATA ITEM NAME                     : PR01NAME
REQUIRED SOURCE DATA ITEM NAME                     : *FILL=10
REQUIRED SOURCE DATA ITEM NAME                     : PR01LKPR
REQUIRED SOURCE DATA ITEM NAME                     : PR01DATA
REQUIRED SOURCE DATA ITEM NAME                     : END.
DO YOU WANT TO COPY ALL TARGET DATA ITEMS          ? Y



***** Target data item is                         : PR01CTRL
***** Target data item is                         : PR01NAME
***** Target data item is                         : PR01LKPR
***** Target data item is                         : PR01DATA
***** Target data item is                         : END.
```

The following figure shows this sequence of events. The data items are unloaded from the existing PR01 file into the work file. The data item PR01NAME is padded with 10 bytes containing spaces.

NOTE

You can add, delete, and expand data items in one data set in a single unload/reload operation.



PR01CTRL	PR01NAME		PR01LKPR	PR01DATA
F610	J A ADAMS	10 byte spaces	*xxxxxxxx*	20 byte spaces
M259	D N HOPKINS	10 byte spaces	*yyyyyyyy*	20 byte spaces
F901	T L SMITH	10 byte spaces	*zzzzzzzz*	20 byte spaces

Executing DBA utilities

The last prompt that the utilities displays is the “DO YOU WANT TO RUN THE BACKGROUND TASK (Y/S/N)?” prompt.

You can execute DBA utilities in the following ways, depending on how you respond to this prompt:

- Execute them immediately from DBA after you have responded to all the prompts (Y)
- Save your responses and execute them later at the command level (S)
- Save your responses and execute them later from DBA (N)

Executing DBA utilities immediately after responding to the prompts

VMS

Enter Y to execute utilities immediately. By entering Y, DBA immediately submits the process to batch and returns control to your terminal. If you submit the process to batch, the utilities create a log file in your root directory. By default, the log file is called PARAMT.LOG although you may have specified a different file name in response to a previous utilities prompt. When the batch process is completed, the log file is printed on SYS$PRINT and then deleted unless your batch queue’s characteristics specify otherwise.

UNIX

Enter Y to execute utilities immediately. By entering Y, DBA immediately creates a background process and returns control to your terminal. When you submit it as a background process, the utilities create a log file in your current directory. By default, the log file is called paramt.log, although you may have specified a different file name in response to a previous utilities prompt.

Saving your responses and executing utilities later at the command level

Enter S to save your responses so you can execute the utilities later at the command level. When you enter S, the utilities create a set of parameter files in your default directory. The parameter files contain the information you gave DBA during the session and control information used to execute the utilities from the command level. By default, these parameter files take a file name of PARAMT (VMS) or paramt (UNIX) and a file type of PR*n* where *n* is a number from 1 to 5. The number of parameter files created depends on the utility selected. If you specified a different file type in response to a previous prompt, the parameter files will use that file type.

When you execute the utilities later at the command level, you can either execute them online or send them to the batch queue (VMS) or as a background process (UNIX).

To execute the parameter files online, enter:

VMS
```
$ @PARAMT.PR1
```

UNIX
```
paramt.pr1
```

To submit the process to the batch queue (VMS) or as a background process (UNIX), enter:

VMS
```
$SUBMIT PARAMT.PR1
```

UNIX
```
paramt.pr1 >>paramt.log 2>>paramt.log &
```

Saving your responses and executing utilities later from DBA

Enter N to save your responses in a saved answer file and exit DBA utilities.

NOTE

To save your responses so you can execute utilities later from DBA, you must have specified a file name in response to the “IF YOU WANT A SAVED ANSWER FILE, PLEASE SPECIFY NAME:” prompt, which DBA displays at the beginning of the utilities session.

To process the saved answer file later from DBA, select the Utilities function from the Administration Functions menu and specify the name of the saved answer file in response to the “EXECUTE A COMMAND PROCEDURE OR HIT RETURN:” prompt. The following shows the format for the saved answer file name:

VMS `@` *`saved-answer-file`*

UNIX *`saved-answer-file`*

Unloading and reloading a related data set under VMS

To unload and reload a related data set under VMS, you can use the standard procedures for unloading and reloading a data set, as discussed in "Unloading and reloading a primary data set" on page 60, or you can use an alternate method called high-speed secondary linkpath reloading. If you are unloading and reloading a CONTROL MRPDBO database, please refer to the pertinent CONTROL manual for the sort sequence.

Using standard unload/reload processing

The steps involved in unloading and reloading a related data set are the same as those needed to unload and reload a primary data set, namely:

1. Rename the source database description file specification (see "Renaming the database file specification" on page 50).

2. Modify the physical characteristics of the data set to be unloaded and reloaded (see "Modifying the data set's physical file specification" on page 51).

3. Validate and compile the target database description (see "Validating and compiling the new database description" on page 55).

4. If the target database for the Unload/Reload utility is a newly-created database and a task log or a system log was defined for it, format each log (see "Formatting the task and system logs" on page 58).

5. Follow the procedures in "Unloading and reloading a primary data set" on page 60 for unloading and reloading a primary data set with one difference: choose option 2, UNLOAD/RELOAD RELATED DATA SET at the Process Selection menu.

Using high-speed secondary linkpath reloading

You can accelerate the utility processing of related data sets by modifying the database to use the high-speed secondary linkpath reloading routine.

The Load utility loads a related data set in two passes:

1. It adds data records and establishes the primary linkpath. (The definition of the associated primary data set must be identical on the source- and target-database descriptions.)

2. It establishes the secondary linkpaths.

The high-speed secondary linkpath reloading routine saves processing during the second pass of the background task. When using this routine, temporarily change secondary linkpaths to data item names through DBA so that the utility ignores them.

The high-speed secondary linkpath routine supports up to 20 secondary linkpaths. Even if a related data set contains more than 20 secondary linkpaths, only 20 can be modified to data item names. The DBA utilities establish any linkpaths not temporarily modified to data item names during the initial reload pass. However, this is a less efficient method of establishing linkpaths when compared to high-speed reloading.

NOTE

To run DBA utilities on a related data set containing coded records, the same linkpath must be present in every coded record. Do not attempt to modify data items or linkpaths in the coded portion of the related record. Also, when adding data items to the redefined portion of more than one record code, unload and reload before adding data items to the redefined portion of another record code. Do not attempt to add data items to the redefined portion of more than one record code with no intervening unload and reload.

Perform the following procedures to use the high-speed secondary linkpath reloader:

NOTE

Procedures that require further explanation are in boldface. You can find more information for the procedures in boldface after the following list.

1. Perform the following:

   a. Rename the source database description file specification (see "Renaming the database file specification" on page 50).

   b. Modify the physical characteristics of the data set to be unloaded and reloaded (see "Modifying the data set's physical file specification" on page 51).

   c. Validate and compile the target database description (see "Validating and compiling the new database description" on page 55).

2. Change the names of all secondary linkpaths to data item names. Use any data item names that do not already exist in the data set.

3. Invoke DBA utilities and select the unload/reload related data set function.

4. Identify the database and work files.

5. Specify the related data items to be unloaded from the source database and reloaded onto the target database.

6. Use DBA to reset the data item names to their original linkpath definitions.

7. If you use system logging on the reloaded database, back up the database files and format a new system log.

Change secondary linkpath names to data item names

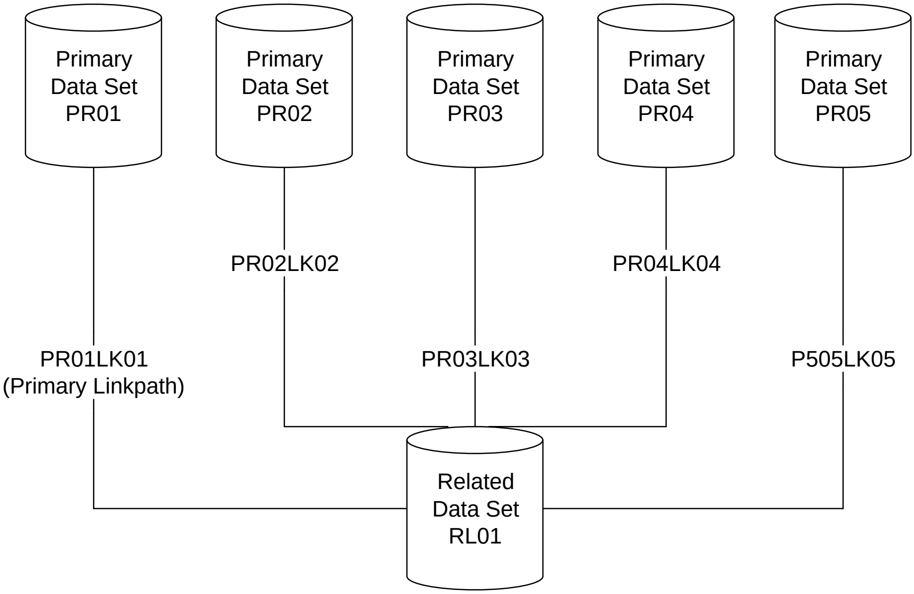
The following example shows the changes required to the linkpath definitions to allow the high-speed, secondary linkpath reloading routine to operate. The linkpath names PR02LK02, PR03LK03, and PR04LK04 are replaced by the data item names RL01AAAA, RL01BBBB, and RL01CCCC. These names are specified as source and target data items.

NOTE

You must replace the linkpath names after the reload is complete so normal processing may continue.

Data items

```
RL01RKEY
RL01PR01
PR01LK01      (Primary linkpath established by the PDM)
RL01PR02
RL01AAAA      (Changed from PR02LK02 for high-speed reload)
RL01PR03
RL01BBBB      (Changed from PR03LK03 for high-speed reload)
RL01DATA
RL01CD01
RL01DA06
RL01PR04
RL01CCCC      (Changed from PR04LK04 for high-speed reload)
RL01CD02
RL01PR05
PR05LK05      (Unchanged because a data item follows the
RL01DA02      linkpath in coded record)
END.
```


Primary
Data Set
PR01
Primary
Data Set
PR02
Primary
Data Set
PR03
Primary
Data Set
PR04
Primary
Data Set
PR05
PR02LK02
PR04LK04
PR01LK01
(Primary Linkpath)
PR03LK03
P505LK05
Related
Data Set
RL01

Identify the database and work files

- Specify the source database description details in response to the prompts
- Answer YES to the “DO YOU WANT TO PRESERVE THE LOGICAL SEQUENCE OF RECORDS ON THE PRIMARY LINKPATH?” prompt
- Specify the target database description details in response to the prompts

The following screen shows sample user input. All the fields shown on this screen may not display. For example if you do not specify a tape drive, tape label fields are not displayed. In addition, if the related data set does not have secondary linkpaths, all associated secondary link prompts do not display. Default values and sample user input are included for illustration.

```
SOURCE DATABASE DESCRIPTION FILE-SPEC         : UTILDB.SRC
SOURCE DATABASE NAME                          : UTILDB
SOURCE DATA SET NAME                          : RL01
SOURCE DATABASE PASSWORD                      :
DO YOU WANT TO PRESERVE THE LOGICAL SEQUENCE
OF RECORDS ON THE PRIMARY LINKPATH            ? Y
PRIMARY LINKPATH NAME                         : PR01LK01
TARGET DATABASE DESCRIPTION FILE-SPEC         : UTILDB.MOD
TARGET DATABASE NAME                          : UTILDB
TARGET DATA SET NAME                          : RL01
TARGET DATABASE PASSWORD                      :
ENTER NAME OF SECONDARY LINKPATHS
TO BE BUILT BY THE RELOADER                   : RL01AAAA,PR02LK02
ENTER NAME OF SECONDARY LINKPATHS
TO BE BUILT BY THE RELOADER                   : END.
WORK FILE SPECIFICATION                       : WORKOUT.TMP
WORK FILE TAPE LABEL                          : WRKOUT
SECONDARY LINKPATH WORK FILE SPECIFICATION    : WORKSEC.TMP
SECONDARY LINKPATH WORK FILE TAPE LABEL       : WRKSEC
SORT OUTPUT FILE SPECIFICATION                : SORTOUT.TMP
SORT OUTPUT FILE TAPE LABEL                   : SRTOUT
SECONDARY SORT OUTPUT FILE SPECIFICATION      : SORTSEC.TMP
SECONDARY SORT OUTPUT FILE TAPE LABEL         : SRTSEC
DEVICE FOR SORT WORK FILES                    :
NUMBER OF SORT WORK FILES                     :
```

SOURCE DATABASE DESCRIPTION FILE-SPEC

Description *Required*. Specifies the file specification of the source database description.

Format Valid VMS file specification

Considerations

- This file specification should match the source file specification as supplied with the RENAME command.
- If you omit a valid file specification, DBA displays an error message and reprompts for the password.
- If you omit the file type, it defaults to .MOD.

SOURCE DATABASE NAME

Description *Required*. Specifies the logical name of the database containing the data set to be unloaded.

Format 6 alphanumeric characters

SOURCE DATA SET NAME

Description *Required*. Specifies the related data set to be unloaded.

Format 4 alphanumeric characters

Consideration This data set must be defined in the source database description.

SOURCE DATABASE PASSWORD

Description *Required* if the database description used a password. This name must match that password.

Format 6 alphanumeric characters

Considerations

- ♦ If you do not enter the correct password, DBA displays an error message and reprompts for the password.
- ♦ If the database description did not use a password, press RETURN.
- ♦ If the password is less than 6 characters, enter spaces to fill 6 characters.

DO YOU WANT TO PRESERVE THE LOGICAL SEQUENCE OF RECORDS ON THE PRIMARY LINKPATH?

Description *Required*. Specifies whether to preserve the logical sequence of records on the primary linkpath during the reorganization process.

Default Y

Options		
	Y	Preserves the logical sequence
	N	Not necessary to preserve the logical sequence

Consideration When performing an unload/reload operation on the Directory database files UDD2 or UDD3, you must answer YES to this prompt. If you do not answer YES, CSIDBA and other components will not be able to use your directory database.

PRIMARY LINKPATH NAME

Description *Required*. Specifies the name of the primary linkpath through this related data set.

Format 8 alphanumeric characters

Considerations

- This entry controls the unloading (if the logical sequence of records is maintained) and reloading of the data records.
- If you change the primary linkpath during reorganization, specify the new linkpath name.
- This linkpath must be present in every record on the related data set.

TARGET DATABASE DESCRIPTION FILE-SPEC

Description *Required*. Specifies the file specification of the target database.

Format Valid VMS file specification

Considerations

- This file specification should match the target file specification of the new database description.
- If you omit a valid file specification, it defaults to the current directory.
- If you omit the file type, it defaults to .MOD.

TARGET DATABASE NAME

Description *Required*. Specifies the logical name of the database containing the data set to be loaded.

Format 6 alphanumeric characters

TARGET DATA SET NAME

Description *Required*. Specifies the related data set to be loaded.

Format 4 alphanumeric characters

Consideration This data set must be defined in the target database description.

TARGET DATABASE PASSWORD

Description *Required* if the database description name used a password.

Format 6 alphanumeric characters

Considerations

- If you do not enter the correct name, DBA displays an error message and reprompts for the password.
- If the database description did not use a password, press RETURN.
- If the password is less than 6 characters, enter spaces to fill 6 characters.

ENTER NAME OF SECONDARY LINKPATHS TO BE BUILT BY THE RELOADER

Description *Optional*. Specifies a data item name that replaces a linkpath for the high-speed reloader and the linkpath being built.

Format

rrrrxxxx	Represents the data item name that temporarily replaces the linkpath, for example, RL01AAAA (see the illustration earlier in this section).
*pppp*LK*xx*	Represents the linkpath name being rebuilt, for example, PR02LK02 (see the illustration earlier in this section).

Considerations

- You can specify a maximum of twenty names.
- Separate each name with a comma.
- Enter END. as the last value or as the first value if all secondary linkpaths are to be handled by the PDM.

WORK FILE SPECIFICATION

Description *Optional*. Specifies the file specification of the work file used to hold the data from the unloaded data set and spaces for new data items.

Default WORKOUT.TMP

Format Valid VMS file specification

WORK FILE TAPE LABEL

Restriction Displayed only if the work file specification indicates a tape drive.

Description *Optional*. Specifies the label of the tape to contain the work file.

Default WRKOUT

Format 6 alphanumeric characters

SECONDARY LINKPATH WORK FILE SPECIFICATION

Description *Optional*. Specifies a sequential work file used to hold the secondary linkpath data.

Default WORKSEC.TMP

Format Valid VMS file specification

SECONDARY LINKPATH WORK FILE TAPE LABEL

Restriction Displayed only if the secondary linkage work file specification indicates a tape drive.

Description *Optional*. Specifies the label of the tape to contain the secondary linkage work file

Default WRKSEC

Format 6 alphanumeric characters

SORT OUTPUT FILE SPECIFICATION

Description *Optional*. Specifies a sequential sort output file used when the logical sequence of records on the primary linkpath need not be preserved.

Default SORTOUT.TMP

Format Valid VMS file specification

SORT OUTPUT FILE TAPE LABEL

Restriction Displayed only if the sort output file specifies a tape drive.

Description *Optional*. Specifies the label of the tape to contain the sort output file.

Default SRTOUT

Format 6 alphanumeric characters

SECONDARY SORT OUTPUT FILE SPECIFICATION

Description *Optional*. Specifies a secondary, sequential sort output file used when the logical sequence of records on the primary linkpath need not be preserved.

Default SORTSEC.TMP

Format Valid VMS file specification

SECONDARY SORT OUTPUT FILE TAPE LABEL

Restriction Displayed only if the secondary sort output file specifies a tape drive.

Description *Optional*. Specifies the label of the tape to contain the secondary sort output file.

Default SRTSEC

Format 6 alphanumeric characters

DEVICE FOR SORT WORK FILES

Description *Optional*. Specifies the device to be used for the sort work files.

Default The device defined when the DEC SORT utility was installed.

Format 5 alphanumeric characters

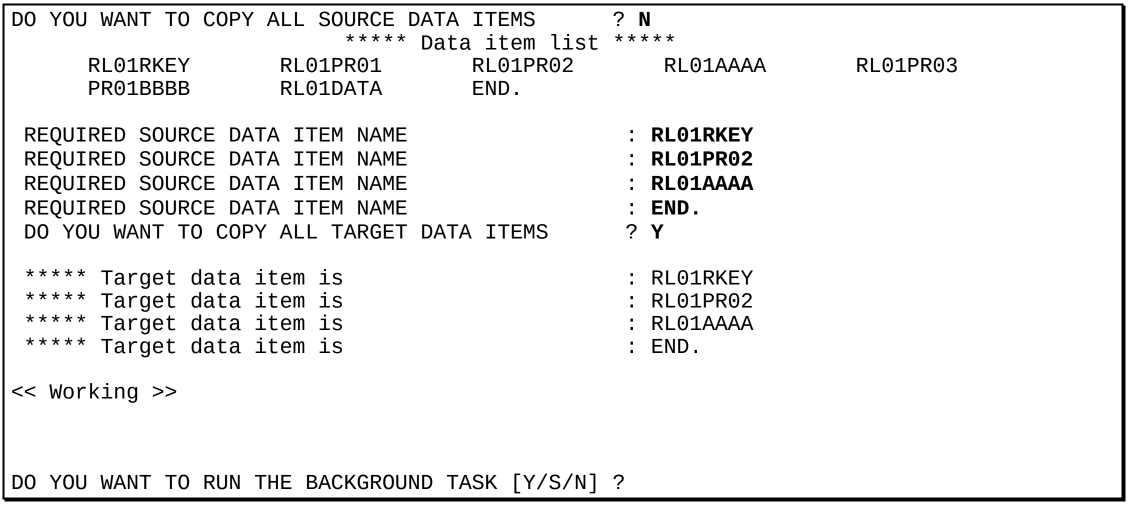
NUMBER OF SORT WORK FILES

Description *Optional*. Specifies the number of work files to be used by the sort routines.

Default The number of files defined when the DEC SORT utility was installed.

Options 0, 2–10

Specify the related data items to be unloaded from the source database and reloaded onto the target database. After you identify the databases and work files, the DBA utilities prompt you to enter the data items to be unloaded from the source data set and the data items to be loaded onto the target data set. Either execute the background task or store the data on the saved answer file for subsequent execution.



```
DO YOU WANT TO COPY ALL SOURCE DATA ITEMS      ? N
                          ***** Data item list *****
      RL01RKEY       RL01PR01       RL01PR02       RL01AAAA       RL01PR03
      PR01BBBB       RL01DATA       END.

 REQUIRED SOURCE DATA ITEM NAME                  : RL01RKEY
 REQUIRED SOURCE DATA ITEM NAME                  : RL01PR02
 REQUIRED SOURCE DATA ITEM NAME                  : RL01AAAA
 REQUIRED SOURCE DATA ITEM NAME                  : END.
 DO YOU WANT TO COPY ALL TARGET DATA ITEMS      ? Y

 ***** Target data item is                       : RL01RKEY
 ***** Target data item is                       : RL01PR02
 ***** Target data item is                       : RL01AAAA
 ***** Target data item is                       : END.

<< Working >>



DO YOU WANT TO RUN THE BACKGROUND TASK [Y/S/N] ?
```

DO YOU WANT TO COPY ALL SOURCE DATA ITEMS?

Description *Required*. Specifies whether to unload all data items on the source data set.

Default Y

Options Y Copies all items

N Does not copy all items

Considerations

- If you enter Y, the source data items to be unloaded display. Each data item is preceded by "***** Source data item is." All data items, but not all linkpaths, are unloaded.
- If you enter N, use the displayed list to specify the names of the source data items to be unloaded.

REQUIRED SOURCE DATA ITEM NAME

Restriction Displayed only if previous response was N.

Description *Required*. Specifies each data item to be included on the data set.

Format 8 alphanumeric characters or END.

Considerations

- The first data item must be the name of the control key for the primary linkpath.
- The DBA utilities redisplay this prompt for you to enter up to 998 data items.
- If the source data set contains coded records, the second data item must be the CODE data item (*rrrr*CODE).
- To insert blank positions on the unloaded data set, enter *FILL=*nn*, where *nn* indicates the number of blank positions.
- Enter END. after you specify all required data items.

DO YOU WANT TO COPY ALL TARGET DATA ITEMS?

Description *Required*. Specifies whether to load all data items on the target data set.

Default Y

Options Y Loads all items

N Does not load all items

Considerations

- If you enter Y, the DBA utilities display the target data items to be loaded for your review.
- If you enter N, select the names of the target data items to be loaded. All available data item names are displayed.

REQUIRED TARGET DATA ITEM NAME

Restriction Displayed only if previous response was N.

Description *Required*. Specifies each data item to be included on the data set.

Format 8 alphanumeric characters or END.

Considerations

- The first data item must be the name of the control key for the primary linkpath.
- The DBA utilities redisplay this prompt for you to enter up to 998 data items.
- If the target data set contains coded records, the second data item must be the CODE data item (*rrrr*CODE).
- If you indicated blank positions with the source data items (*FILL=*nn*), enter the data item name for these positions.
- When data items are selectively loaded, the target data items must correspond exactly to the source data items—they must be entered in the same sequence.
- Enter END. after you specify all required data items.

DO YOU WANT TO RUN THE BACKGROUND TASK [Y/S/N]?

Description *Required*. Specifies whether to execute the background task immediately or wait for subsequent execution.

Default N

Options Y Executes now

S Stores parameter files for future execution

N Deletes parameter files

Consideration See “Executing DBA utilities” on page 75 for more information.

Unloading and reloading a related data set under UNIX

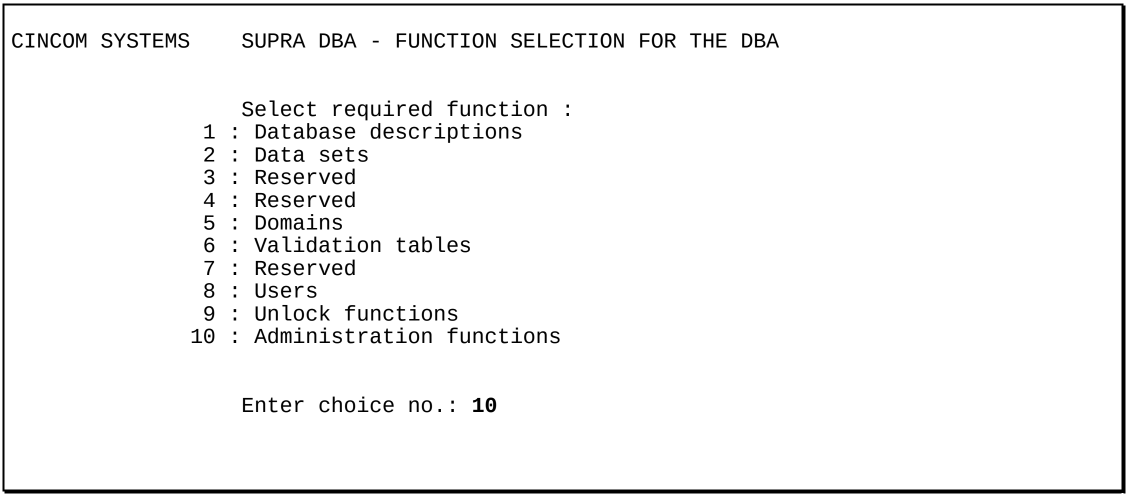
The preliminary steps involved in unloading and reloading a related data set are the same as those needed to unload and reload a primary data set, namely:

- Rename the source database description file specification (see “Renaming the database file specification” on page 50)
- Modify the physical characteristics of the data set to be unloaded and reloaded (see “Modifying the data set’s physical file specification” on page 51)
- Validate and compile the new database description (see “Validating and compiling the new database description” on page 55)
- If the target database for the Unload/Reload utility is a newly created database and a task log or system log was defined for it, format each log (see “Formatting the task and system logs” on page 58)

NOTE

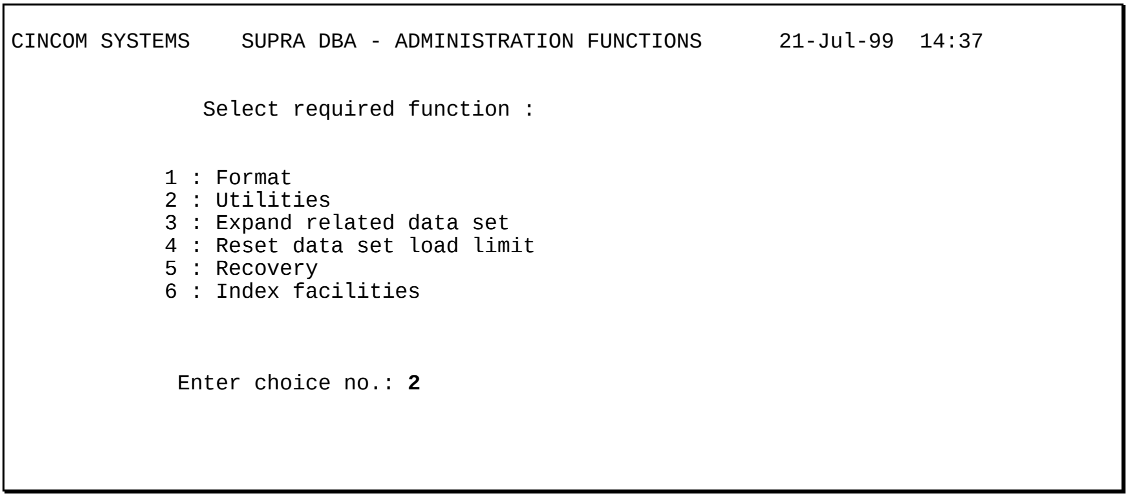
If you use system logging on a reloaded database, ensure that the reload is successful; then back up the database files and format a new system log. Do not allow any users to access the database description until you format a new system log file.

This section uses the related data set RL01 to show how to unload and reload data items in a related data set. To initiate Utilities, select option 10, Administration functions.



```
CINCOM SYSTEMS    SUPRA DBA - FUNCTION SELECTION FOR THE DBA


                  Select required function :
               1 : Database descriptions
               2 : Data sets
               3 : Reserved
               4 : Reserved
               5 : Domains
               6 : Validation tables
               7 : Reserved
               8 : Users
               9 : Unlock functions
              10 : Administration functions


                  Enter choice no.: 10
```

This displays the Administration Functions menu. Select option 2, Utilities.



```
CINCOM SYSTEMS    SUPRA DBA - ADMINISTRATION FUNCTIONS      21-Jul-99  14:37


               Select required function :


            1 : Format
            2 : Utilities
            3 : Expand related data set
            4 : Reset data set load limit
            5 : Recovery
            6 : Index facilities



             Enter choice no.: 2
```

DBA presents the Process Selection menu. Specify a name for the saved answer file if you wish to keep the answers from this session. Specify the file specification prefix for the parameter files unless you want to use the default, PARAMT. Select option 2, UNLOAD/RELOAD RELATED DATA SET.

```
ENTER PATH TO UTLNAT.SEQ :
EXECUTE A COMMAND PROCEDURE OR HIT RETURN :

SUPRA DATABASE ADMINISTRATION UTILITIES V1.0

=============================================

IF YOU WANT A SAVED ANSWER FILE, PLEASE SPECIFY NAME :
 ENTER FILE SPECIFICATION PREFIX FOR PARAMETER FILES  : PARAMT

PROCESS SELECTION
 ===========================================================
 - UNLOAD/RELOAD PRIMARY DATA SET                         (1)
 - UNLOAD/RELOAD RELATED DATA SET                         (2)
 - ADD RECORDS TO PRIMARY DATA SET                        (3)
 - ADD RECORDS TO RELATED DATA SET                        (4)
 - DELETE RECORDS FROM PRIMARY DATA SET                   (5)
 - DELETE RECORDS FROM RELATED DATA SET                   (6)
 - PRODUCE DATABASE STATISTICS                            (7)
 ===========================================================
 ENTER CORRESPONDING NUMBER                              : 2
```

The DBA utilities then prompts for source and target database/data set identification, and logical sequence and primary linkpath to be unloaded/reloaded. If the related data set has secondary links, all secondary links are selected and reloaded automatically by the reloader.

```
SOURCE DATABASE DESCRIPTION FILE_SPEC      : /path1/path2/sampdb.src
SOURCE DATABASE NAME                       : sampdb
SOURCE DATA SET NAME                       : RL01
SOURCE DATABASE PASSWORD                   :
DO YOU WANT TO PRESERVE THE LOGICAL SEQUENCE
OF RECORDS ON THE PRIMARY LINKPATH?        : N
PRIMARY LINKPATH NAME                      : PR01LK01
TARGET DATABASE DESCRIPTION FILE-SPEC      : /path1/path2/sampdb.mod
TARGET DATABASE NAME                       : sampdb
TARGET DATA SET NAME                       : RL01
TARGET DATABASE PASSWORD                   :
WORK FILE SPECIFICATION                    : workout.tmp
SORT OUTPUT FILE SPECIFICATION             : sortout.tmp
DO YOU WANT TO COPY ALL SOURCE DATA ITEMS  : Y
***** Source data item is                  : RL01CTRL
***** Source data item is                  : RL01CODE
***** Source data item is                  : RL01DATA
DO YOU WANT TO COPY ALL TARGET DATA ITEMS ?: Y
***** Target data item is                  : RL01CTRL
***** Target data item is                  : RL01CODE
***** Target data item is                  : RL01DATA
<< Working >>
DO YOU WANT TO RUN THE BACKGROUND TASK [Y/S/N]? Y
```

SOURCE DATABASE DESCRIPTION FILE-SPEC

Description *Required*. Specifies the file specification of the source database description.

Format Valid UNIX pathname

Considerations

- This file specification should match the source file specification used with the mv command.
- If you omit the directory name, it defaults to the current directory.
- If you omit the file suffix, it defaults to .mod.

SOURCE DATABASE NAME

Description *Required*. Specifies the logical name of the database containing the data set to be unloaded.

Format 6 alphanumeric characters

SOURCE DATA SET NAME

Description *Required*. Specifies the related data set to be unloaded.

Format 4 alphanumeric characters

Consideration This data set must be defined in the source database description and may contain up to 998 data items.

SOURCE DATABASE PASSWORD

Description *Required* if the database description uses a password. This name must match that password.

Format 6 alphanumeric characters

Considerations

- The password does not display.
- If you omit the correct password, DBA displays an error message and reprompts for the password.
- If the database description did not use a password, press RETURN.
- If the password is less than 6 characters, enter spaces to fill 6 characters.

DO YOU WANT TO PRESERVE THE LOGICAL SEQUENCE OF RECORDS ON THE PRIMARY LINKPATH?

Description *Required*. Specifies whether to preserve the logical sequence of records on the primary linkpath during the unloading process.

Default Y

Options

Y Preserves the logical sequence. This option causes the unloader to serially read the primary data set and unload any control key for which related records exist from the related data set. The unload process reads the related data set in sequence via the primary link from the first related record on the link to the last related record on the link. You cannot use this option if the links on the related data set are corrupt or invalid.

N Not necessary to preserve the logical sequence. This option causes the unloader to serially read the related data set and unload data items without the use of the primary link. Records are unloaded in the physical order in which they appear in the related data set. The reloader will sort the records by primary key's home location prior to reloading the data set to ensure that all records with the same key are added together.

Consideration When unloading and reloading the Directory database files UDD2 or UDD3, you must answer YES to this prompt. If you do not answer YES, CSIDBA and other components will not be able to use your directory database.

PRIMARY LINKPATH NAME

Description *Required*. Specifies the name of the primary linkpath through this related data set.

Format 8 alphabetic characters

Considerations

- This entry controls the unloading (if the logical sequence of records is maintained) and reloading of the data records.
- This linkpath must be present in every record on the related data set.
- Secondary linkpaths are automatically selected and maintained for you by the reload process.

TARGET DATABASE DESCRIPTION FILE-SPEC

Description *Required*. Specifies the file specification of the target database.

Format Valid UNIX pathname

Considerations

- This file specification should match that of the new database description.
- If you omit the /directory, it defaults to the current directory.
- If you omit the file suffix, it defaults to .mod.

TARGET DATABASE NAME

Description *Required*. Specifies the logical name of the database containing the data set to be loaded.

Format 6 alphanumeric characters

TARGET DATA SET NAME

Description *Required*. Specifies the related data set to be loaded.

Format 4 alphanumeric characters

Consideration This data set must be defined in the target database description.

TARGET DATABASE PASSWORD

Description *Required* if the database description used a password. This name must match that password.

Format 6 alphanumeric characters

Considerations

- The password does not display, but if you omit the correct password, DBA displays an error message and reprompts for the password.
- If the database description did not use a password, press RETURN.
- If the password is less than 6 characters, enter spaces to fill 6 characters.

WORK FILE SPECIFICATION

Description *Optional*. Specifies the file specification of the work file used to hold the data from the unloaded data set and spaces for new data items.

Default workout.tmp

Format Valid UNIX pathname or tape device name such as /dev/rmt/0m

***** Default work file specification is

Restriction Displays only if you do not enter a value for the temporary work file specification.

Description Displays the default value for the temporary work file specification, workout.tmp

SORT OUTPUT FILE SPECIFICATION

Description *Optional*. Specifies a sequential sort output file used when the logical sequence of records on the primary linkpath is not preserved.

Default workout.tmp

Format Valid UNIX pathname or tape device name such as /dev/rmt/0m. The tape device can be the same for the work file specification and the sort output file specification.

DO YOU WANT TO COPY ALL SOURCE DATA ITEMS?

Description *Required*. Specifies whether to unload all data items on the source data set.

Default Y

Options

Y	Copies all items
N	Does not copy all items

Considerations

- If you enter Y, the source data items to be unloaded display. Each data item is preceded by "***** Source data item is". All data items, but not all linkpaths, are unloaded.
- If you enter N, use the displayed list to specify the names of the source data items to be unloaded.

DO YOU WANT TO COPY ALL TARGET DATA ITEMS?

Description *Required*. Specifies whether to load all data items on the target data set.

Default Y

Options Y Loads all items

N Does not load all items

Considerations

- If you enter Y, the DBA utilities display the target data items to be loaded for your review.
- If you enter N, select the names of the target data items to be loaded. All available data-item names are displayed.

DO YOU WANT TO RUN THE BACKGROUND TASK [Y/S/N]?

Description *Required*. Specifies whether to execute the background task immediately or wait for subsequent execution.

Default N

Options Y Executes now

S Stores parameter files for future execution

N Deletes parameter files

Consideration See "Executing DBA utilities" on page 75 for more information.

Recompiling all connected databases

When you run the Unload/Reload utility on a data set that is connected to more than one database via the Multiple Physical Database facility, you must recompile all connected database descriptions. The data-set details in any database connected to the modified data set will not be accurate after unloading and reloading.

If the connected databases also use a different physical file for the data set, you must also rerun the Unload/Reload utility. The following examples illustrate these situations:

Example 1

You wish to run the Unload/Reload utility on the primary data set PR01, which is connected to the two databases UTILDB and TESTDB. The physical file specification for the data set PR01 is the same for both databases: PR01.SRC.

You recompile UTILDB and run the Unload/Reload utility to unload the data from PR01.SRC, reloading it into PR01.NEW. The recompiled database description for UTILDB uses the new file PR01.NEW, which contains the reloaded data records for PR01.

The database description for TESTDB is now invalid because of the physical changes you made to the data set PR01 (by adding, deleting, or expanding data items). In addition, the database description for TESTDB still uses the unmodified file, PR01.SRC. Because both databases used the same file for PR01, they share the data records in that file. Therefore, you need to recompile TESTDB. Before compilation, you must update the database description to use the new file PR01.NEW.

Example 2

You wish to run the Unload/Reload utility on the primary data set PR01, which is connected to the two databases UTILDB and PR0CDB. However, the two databases use different physical file specifications for PR01. UTILDB uses the file PR01.SRC; PR0CDB uses the file PR01.DAT. The records held in PR01 may, therefore, be different for each database.

As in Example 1, you recompile UTILDB and run the Unload/Reload utility to unload the data from PR01.SRC, and then reload it into PR01.NEW. The recompiled database description for UTILDB uses the new file PR01.NEW, which contains the reloaded data records for the data set PR01.

The database description for PROCDB is now invalid because of the physical changes you made to the data set PR01 (by adding, deleting, or expanding data items). You must not only recompile the database PROCDB but also rerun the Unload/Reload utility, specifying a new physical file to hold the records from PR01.DAT.

If you only recompile the database PROCDB as in Example 1, the database description will be updated to reflect the new record layout. This record layout will not match the data records held in the physical file PR01.DAT, and any attempt to access the file will have unpredictable results.

Formatting a new system log for a reloaded database

If you use system logging on a reloaded database, first ensure that the reload has completed successfully; then do the following:

1. Back up the database files.

2. Format a new system log.

Do not allow any users to access the database description until you format a new system log file.

Using the Add utility

Use the Add utility to add records to existing data sets or to newly formatted data sets. The Add utility is a quick method to insert a large volume of data into a data set. You can add records to primary and related data sets; however, the Add utility prompts you to specify the primary linkpath when adding records to related data sets.

Preparing to use the Add utility

Before you invoke the Add utility, create an input data file containing all the records you are adding to the data set. You can create this file using a standard text editor. The input data file must be available before you run the Add utility.

Records are added in the sequence in which they appear in the input file. The DBA utilities consider each record a string of characters that must match the list of data items specified to the utility. Calculate the logical record length of the input records by adding the lengths of all of the data items being added. Ensure the length of each record on the input data file is the same as the logical record's length previously calculated.

UNIX

Records in the input file must be contiguous with no additional data or control characters separating each record. For small input files (less than 2K), the join command of a standard text editor can be used to obtain contiguous records (in vi or ex use:1,$j!). For larger input files, record separators can be removed with the tr filter. For example, to remove newline characters use:

```
cat file | tr -d “\012” > newfile
```

If the input data itself contains control characters, possibly in the form of binary data, then a hex editor or user-written program must be used to produce an input file in the required format.

Adding records to a primary data set

Select option 3, ADD RECORDS TO PRIMARY DATA SET, from the Process Selection menu. DBA returns the following prompts, depending on your responses:

```
DATABASE DESCRIPTION FILE-SPEC                    :
DATABASE NAME                                     :
PRIMARY DATA SET NAME                             :
DATABASE PASSWORD                                 :
DO YOU WANT TO CREATE ALL DATA ITEMS              :


***** Source data item is                         :
***** Source data item is                         :
***** Source data item is                         :


INPUT DATA FILE SPECIFICATION                     :



DO YOU WANT TO RUN THE BACKGROUND TASK [Y/S/N] ? :
```

DATABASE DESCRIPTION FILE-SPEC

Description *Required*. Specifies the name of the file containing the database description.

Format VMS Valid file specification

UNIX Valid pathname

DATABASE NAME

Description *Required*. Specifies the name of the database containing the data set.

Format 6 alphanumeric characters

PRIMARY DATA SET NAME

Description *Required*. Specifies the name of the primary data set to which the records will be added.

Format 4 alphanumeric characters

DATABASE PASSWORD

Description *Required* if the database description uses a password. This name must match that password.

Format 6 alphanumeric characters

Considerations

- If you do not enter the correct password, DBA displays an error message and reprompts for the password.
- If the database description did not use a password, press RETURN.
- If the password is less than 6 characters, enter spaces to fill 6 characters.

DO YOU WANT TO CREATE ALL DATA ITEMS

Description *Required*. Specifies whether you are creating all data items on the primary data set or only selected data items.

Options Y Creates all data items

N Selects data items

Considerations

- If you enter Y, the source data items display. Each data item is preceded by “****Source data item is”.
- If you enter N, use the displayed list to specify the names of the source data items you wish to create.

REQUIRED DATA ITEM NAME

Restriction Displayed only if the previous response was N.

Description *Required*. Allows you to specify each data item you want to include on the data set.

Format 8 alphanumeric characters

Considerations

- You must specify the items in the order in which they are arranged in the input file.
- The first data item you specify must be the name of the control key for the primary linkpath for the related record. This must also be the first field in the input file.
- If the data set contains coded records, the second item you specify must be the record code data item. This must be the second field in the input file. Specify the coded data portion of the record as one data item.
- The DBA redisplays the prompt for you to enter up to 998 data items in the order in which they appear in the input file.
- Enter END. after you specify all data items.

INPUT DATA FILE SPECIFICATION

Description *Required*. Specifies the name of the input data file containing the data to be added to the data set.

Format **VMS** Valid file specification

UNIX Valid pathname

TAPE LABEL **VMS**

Restriction Displays only if the file specification indicates a tape drive.

Description *Optional*. Specifies the label of the tape containing the input data file.

Default UTLTAP

Format 6 alphanumeric characters

DO YOU WANT TO RUN THE BACKGROUND TASK [Y/S/N]?

Description *Required*. Specifies whether to execute the background task immediately or wait for subsequent execution.

Default N

Options Y Executes now

S Stores parameter files for future execution

N Deletes parameter files

Consideration See “Executing DBA utilities” on page 75 for additional information.

Adding records to a related data set

Select option 4, ADD RECORDS TO RELATED DATA SET, from the Process Selection menu. DBA returns the following prompts, depending on your responses:

```
DATABASE DESCRIPTION FILE-SPEC                    :
DATABASE NAME                                     :
RELATED DATA SET NAME                             :
DATABASE PASSWORD                                 :
PRIMARY LINKPATH NAME                             :
DO YOU WANT TO CREATE ALL DATA ITEMS              :
REQUIRED DATA ITEM NAME                           :


    ***** data item list *****


INPUT DATA FILE SPECIFICATION                     :




DO YOU WANT TO RUN THE BACKGROUND TASK [Y/S/N] ?  :
```

DATABASE DESCRIPTION FILE-SPEC

Description *Required*. Specifies the name of the file containing the database description.

Format VMS Valid file specification

UNIX Valid pathname

DATABASE NAME

Description *Required*. Specifies the name of the database containing the data set.

Format 6 alphanumeric characters

RELATED DATA SET NAME

Description *Required*. Specifies the name of the related data set to which the records will be added.

Format 4 alphanumeric characters

DATABASE PASSWORD

Description *Required* if the database description used a password. This name must match that password.

Format 6 alphanumeric characters

Considerations

- If you do not enter the correct password, DBA displays an error message and reprompts for the correct password.
- If the database description did not use a password, press RETURN.
- If the password is less than 6 characters, enter spaces to fill 6 characters.

PRIMARY LINKPATH NAME

Description *Required*. Specifies the name of the primary linkpath to this related data set.

Format 8 alphanumeric characters

DO YOU WANT TO CREATE ALL DATA ITEMS

Description *Required*. Specifies whether you are creating all data items on the related data set or only selected data items.

Options

Y	Creates all data items
N	Selects data items

Considerations

- ♦ If you enter Y, the source data items display. Each data item is preceded by "***** Source data item is".
- ♦ If you enter N, use the displayed list to specify the names of the source data items you wish to create.

REQUIRED DATA ITEM NAME

Restriction Displayed only if the previous response was N.

Description *Required*. Specifies each data item you want to include on the data set.

Format 8 alphanumeric characters or END.

Considerations

- ♦ The first data item you specify must be the name of the control key on the related record.
- ♦ If the data set contains coded records, the second item you specify must be the record code data item. Specify the coded data portion as one data item.
- ♦ The DBA utilities redisplay this prompt for you to enter up to 998 data items.
- ♦ Enter END. after you specify all data items you require.

INPUT DATA FILE SPECIFICATION

Description *Required*. Specifies the name of the input data file containing the data to be added to the data set.

Format

VMS Valid file specification

UNIX Valid pathname

TAPE LABEL VMS

Restriction Displayed only if the file specification indicates a tape drive.

Description *Optional*. Specifies the label of the tape containing the input data file.

Default UTLTAP

Format 6 alphanumeric characters

DO YOU WANT TO RUN THE BACKGROUND TASK [Y/S/N]?

Description *Required*. Specifies whether to execute the background task immediately or wait for subsequent execution.

Default N

Options Y Executes now

S Stores parameter files for future execution

N Deletes parameter files

Consideration See "Executing DBA utilities" on page 75 for more information.

Using the Delete utility

Use the Delete utility to remove records from primary data sets, and to remove lists of records from related data sets.

Preparing to use the Delete utility

Before you invoke the Delete utility, create an input data file containing the key of each record you want to delete. You can create this file using a standard text editor. Start the key in position 1 of each record. The input data file must be available before you invoke the Delete utility.

UNIX

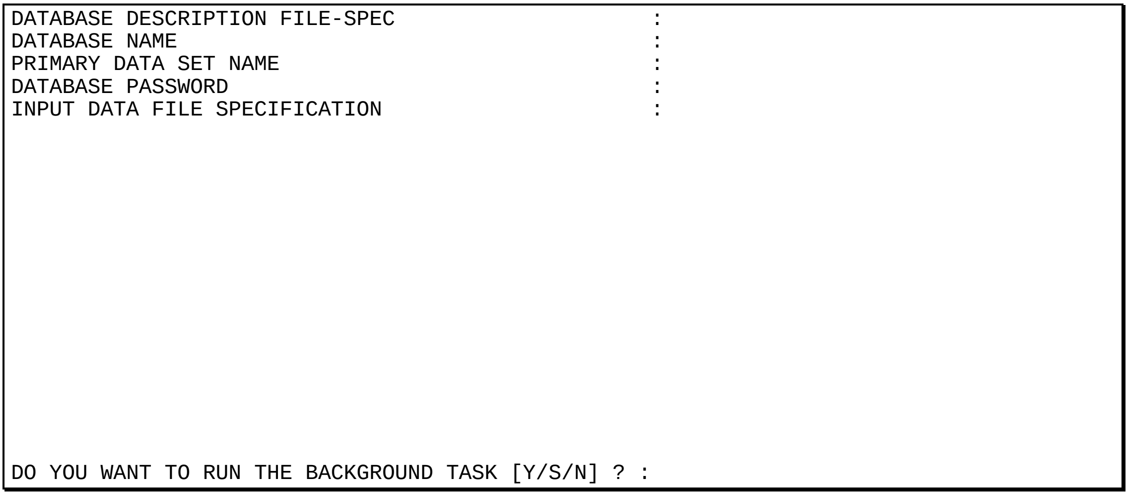
Records in the input file must be contiguous with no additional data or control characters separating each record. For small input files (less than 2K), the join command of a standard text editor can be used to obtain contiguous records (in vi or ex use:1,$j!). For larger input files, record separators can be removed with the tr filter. For example, to remove newline characters use:

```
cat file | tr -d “\012” > newfile
```

If the input data itself contains control characters, possibly in the form of binary data, then a hex editor or user-written program must be used to produce an input file in the required format.

Deleting records from a primary data set

Select option 5, DELETE RECORDS FROM PRIMARY DATA SET, from the Process Selection menu. DBA returns the following prompts, depending on your responses:



```
DATABASE DESCRIPTION FILE-SPEC                      :
DATABASE NAME                                       :
PRIMARY DATA SET NAME                               :
DATABASE PASSWORD                                   :
INPUT DATA FILE SPECIFICATION                       :















DO YOU WANT TO RUN THE BACKGROUND TASK [Y/S/N] ? :
```

DATABASE DESCRIPTION FILE-SPEC

Description *Required*. Specifies the name of the file containing the database description.

Format VMS Valid file specification

UNIX Valid pathname

DATABASE NAME

Description *Required*. Specifies the name of the database containing the data set.

Format 6 alphanumeric characters

PRIMARY DATA SET NAME

Description *Required*. Specifies the name of the primary data set from which the records will be deleted

Format 4 alphanumeric characters

DATABASE PASSWORD

Description *Required* if the database description used a password. This name must match that password.

Format 6 alphanumeric characters

Considerations

- If you do not enter the correct password, DBA displays an error message and reprompts for the password.
- If the database description did not use a password, press RETURN.
- If the password is less than 6 characters long, enter spaces to fill 6 characters.

INPUT DATA FILE SPECIFICATION

Description *Required*. Specifies the name of the input data set containing the data to be deleted from the data set.

Format VMS Valid file specification

UNIX Valid pathname

TAPE LABEL VMS

Restriction Displayed only if the file specification indicates a tape device.

Description *Optional*. Specifies the tape label containing the input data file.

Default UTLTAP

Format 6 alphanumeric characters

DO YOU WANT TO RUN THE BACKGROUND TASK [Y/S/N]?

Description *Required*. Specifies whether to execute the background task immediately or wait for subsequent execution.

Default N

Options Y Executes now

S Stores parameter files for future execution

N Deletes parameter files

Consideration See “Executing DBA utilities” on page 75 for more information.

Deleting records from a related data set

Select option 6, DELETE RECORDS FROM RELATED DATA SET, from the Process Selection menu. DBA returns the following prompts, depending on your responses:

```
DATABASE DESCRIPTION FILE-SPEC                    :
DATABASE NAME                                     :
RELATED DATA SET NAME                             :
DATABASE PASSWORD                                 :
PRIMARY LINKPATH NAME                             :
DO YOU WISH TO DELETE THE PRIMARY RECORD AS
WELL AS THE LIST OF RELATED RECORDS               :
INPUT DATA FILE SPECIFICATION                     :











DO YOU WANT TO RUN THE BACKGROUND TASK [Y/S/N] ? :
```

DATABASE DESCRIPTION FILE-SPEC

Description *Required*. Specifies the name of the file containing the database description.

Format **VMS** Valid file specification

UNIX Valid pathname

DATABASE NAME

Description *Required*. Specifies the name of the database containing the data set.

Format 6 alphanumeric characters

RELATED DATA SET NAME

Description *Required*. Specifies the name of the related data set from which the lists of records will be deleted.

Format 4 alphanumeric characters

DATABASE PASSWORD

Description *Required* if the database description used a password. This name must match that password.

Format 6 alphanumeric characters

Considerations

- If you do not enter the correct password, DBA displays an error message and reprompts for the password.
- If the database description did not use a password, press RETURN.
- If the password is less than 6 characters, enter spaces to fill 6 characters.

PRIMARY LINKPATH NAME

Description *Required*. Specifies the name of the primary linkpath to this related data set.

Format 8 alphanumeric characters

DO YOU WISH TO DELETE THE PRIMARY RECORD AS WELL AS THE LIST OF RELATED RECORDS?

Description *Required*. Specifies whether to delete the primary data set record corresponding to the list of related records being deleted.

Options Y Deletes primary record

N Does not delete primary record

INPUT DATA FILE SPECIFICATION

Description *Required*. Specifies the name of the input data file containing the data to be deleted from the data set.

Format **VMS** Valid file specification

UNIX Valid pathname

TAPE LABEL VMS

Restriction Displays only if the file specification indicates a tape device.

Description *Optional*. Specifies the label of the tape containing the input data file.

Default UTLTAP

Format 6 alphanumeric characters

DO YOU WANT TO RUN THE BACKGROUND TASK [Y/S/N]?

Description *Required*. Specifies whether to execute the background task immediately or wait for subsequent execution.

Default N

Options Y Executes now

S Stores parameter files for future execution

N Deletes parameter files

Consideration See “Executing DBA utilities” on page 75 for more information.

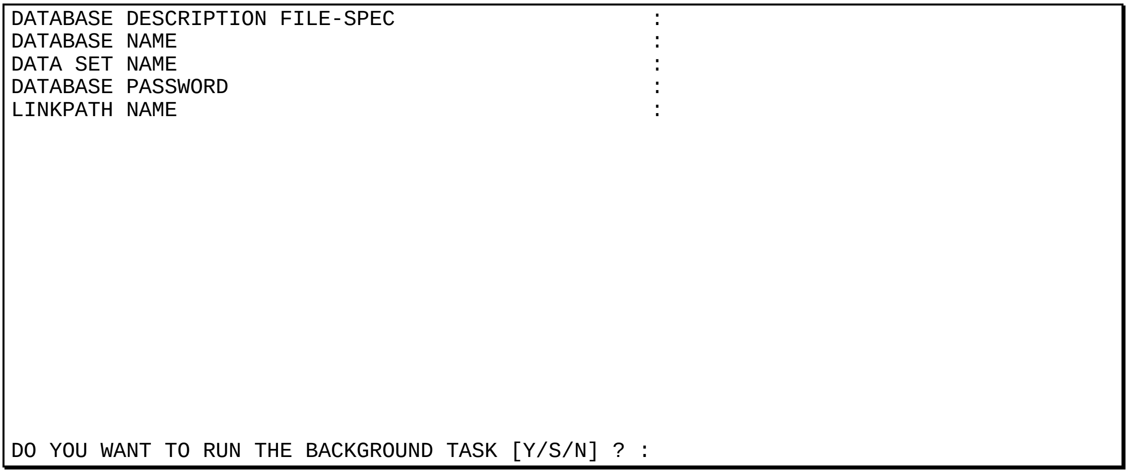
Using the Statistics utility

The Statistics utility analyzes the statistics of the data sets in a database. This utility monitors a production database and produces statistics for:

- A primary data set
- A primary data set and linkpaths
- A related data set, linkpath, and associated primary data set

Producing database statistics

Select option 2, PRODUCE DATABASE STATISTICS, from the Process Selection menu. DBA returns the following prompts so you can select the data set to be analyzed:



```
DATABASE DESCRIPTION FILE-SPEC                        :
DATABASE NAME                                         :
DATA SET NAME                                         :
DATABASE PASSWORD                                     :
LINKPATH NAME                                         :

















DO YOU WANT TO RUN THE BACKGROUND TASK [Y/S/N] ? :
```

DATABASE DESCRIPTION FILE-SPEC

Description *Required*. Specifies the name of the file containing the database description.

Format VMS Valid file specification

UNIX Valid pathname

DATABASE NAME

Description *Required*. Specifies the name of the database containing the data set to be analyzed.

Format 6 alphanumeric characters

DATA SET NAME

Description *Required*. Specifies the data set for which statistics are to be produced.

Format 4 alphanumeric characters

DATABASE PASSWORD

Description *Required* if the database description used a password. This name must match that password.

Format 6 alphanumeric characters

Considerations

- If you do not enter the correct password, DBA displays an error message and reprompts for the password.
- If the database description did not use a password, press RETURN.
- If the password is less than 6 characters, enter spaces to fill 6 characters.

LINKPATH NAME

Description *Conditional*. Optional for the primary data set. Required for the related data set. Specifies a linkpath name for which statistics are to be produced.

Format 8 alphanumeric characters

Considerations

- ♦ If the statistics are being produced for a primary data set, this parameter provides information on the number of primary data-set records that have lists of related records associated with the specified linkpath.
- ♦ If the statistics are being produced for a related data set, this parameter provides data on the condition of this linkpath and on the associated primary data sets.

DO YOU WANT TO RUN THE BACKGROUND TASK [Y/S/N]?

Description *Required*. Specifies whether to execute the background task immediately or wait for subsequent execution.

Default N

Options Y Executes now

S Stores parameter files for future execution

N Deletes parameter files

Consideration See "Executing DBA utilities" on page 75 for more information.

Sample statistics report

The sample statistics report for the primary data set VEND provides some preliminary data set details followed by a summary of performance statistics presented in the table in this section.

```
BEGIN EXECUTION VMS(UNIX)-SUPRA PDM UTILITY
DATABASE-STATISTICS
DATABASE-NAME=XAMPLE
VMS  DBDESC-FILE_SPEC=DRA:[CORP.SUPRA.TEST]XAMPLE.MOD
UNIX DBDESC-FILE-SPEC=/supra/test/xample.mod
DATA-SET-NAME=VEND
LINKPATH=VENDLKAD
KEY=VENDCTRL=0006
PRIMARY-DATA-SET-SIZE
TOTAL-LOGICAL-RECORDS=0000054
LOGICAL-RECORDS-PER-BLOCK=0006
END-PARAMETERS
SIGN ON STATUS =****
VEND DATA SET SUMMARY                TOTAL   PERCENT
_____________________                _____   _______
LOW RRN                                 2
HIGH RRN                               53
NUMBER OF ACTIVE RECORDS        =      33     61.11
RECORDS AT 'HOME' LOCATION      =      25     75.76
RECORDS NOT IN 'HOME' BLOCK     =       1      3.03
NUMBER OF SYNONYM CHAINS        =       7
NUMBER OF SYNONYM RECORDS       =       8     24.24
AVERAGE RECS PER SYNONYM CHAIN  =   1.142
ACTIVE RECS NOT LINKED          =       0      0.00
ACTIVE RECS     LINKED          =      33    100.00

SIGN OFF STATUS=****
STATS
RECORDS IN  =                 33.
RECORDS OUT =                  0.
ERROR COUNT =                  0.
VMS RMS ERRORS  =                  0.
END OF UTILITY JOB
```

Statistics output	**Description**

DATABASE-NAME=*xxxxxx*
DBDESC-FILE-SPEC=*dev:[dir]file.ext* **VMS**
DBDESC-FILE-SPEC=*directory/file.ext* **UNIX**
	6-character database name as defined on the Directory. Full VMS file specification or UNIX pathname to the file containing the database description.

DATA-SET-NAME=*xxxx*	4-character name of the data set for which statistics are produced.
LINKPATH=*xxxxLKxx*	8-character linkpath name for which statistics are produced. (Optional if the data set is primary.)
KEY=*xxxx*CTRL=*nnnn*	8-character name of the primary control key together with its length.
PRIMARY-DATA-SET-*xxxx*	4-character name of the primary data set containing the control key.
TOTAL-LOGICAL-RECORDS=*nnnnnnn*	The maximum number of logical records that can be held.
LOGICAL-RECORDS-PER-BLOCK=*nnnn*	The total number of logical records that can be held in a single block.
LOW RRN	Lowest relative record number in use.
HIGH RRN	Highest relative record number in use.
NUMBER OF ACTIVE RECORDS	The number of active records held, both as an absolute number and as a percentage of total capacity.
RECORDS AT 'HOME' LOCATION	The number of records in their home RRN, both as an absolute number and as a percentage of the total number of active records.

Statistics output	Description
RECORDS NOT IN 'HOME' BLOCK	The number of records not held in the home block, both as an absolute number and as a percentage of the total number of active records.
NUMBER OF SYNONYM CHAINS	Total number of synonym chains in the data set. Refer to the *SUPRA Server PDM System Administration Guide (VMS)*, P25-0130, or the *SUPRA Server PDM System Administration Guide (UNIX)*, P25-0132, for more information on synonyms.
NUMBER OF SYNONYM RECORDS	Total number of synonym records.
AVERAGE RECS PER SYNONYM CHAIN	The average number of records in each synonym chain (number of synonym records divided by the number of synonym chains).
ACTIVE RECS NOT LINKED	The number of active records that have no related records connected to them.
ACTIVE RECS LINKED	The number of active records that have related records connected to them.
RECORDS IN =	Number of records processed.
RECORDS OUT =	Not applicable.
ERROR COUNT =	Number of errors encountered during processing.
RMS ERRORS = **VMS**	Number of RMS errors encountered during processing.

3

Using Fast utilities

You initiate Fast utilities using the VMS command CHANGEDB or the UNIX command csmchangedb.

Fast utilities overview

When using Fast utilities, it is important to understand:

- The physical modifications you can make using Fast utilities
- The steps Fast utilities performs when it executes
- Considerations for running Fast utilities with the Multiple Physical Database Facility

NOTE

Fast utilities requires extra resources; ensure that you have sufficient resources (see “Calculating disk space requirements” on page 133).

Physical modifications you can make using Fast utilities

You can use Fast utilities to make the following physical modifications:

- Increase the total logical records
- Change the number of records per block
- Increase the logical record length
- Change the control interval size
- Change the load limit
- Change or optimize the primary linkpath
- Alter the relative size of the physical files allocated to hold parts of the data set
- Change the shadow file specifications

How Fast utilities operates

When you execute CHANGEDB (VMS) or csmchangedb (UNIX), the Fast utilities program performs these steps:

1. Examines the existing compiled database description for any physical and logical details that were not specified on the command line.

   If the database is active, Fast utilities terminates with the message “Database description file database-name.MOD already in use”. Disable the database by using the SUPRA PDM DISABLE operator command. Refer to the *SUPRA Server PDM System Administration Guide (VMS)*, P25-0130, or the *SUPRA Server PDM System Administration Guide (UNIX)*, P25-0132, for more information on using the PDM DISABLE command.

   If the Database Description Status (as displayed by DBA) is anything other than “O.K.”, Fast utilities terminates with the message “DATABASE NOT COMPILED OK.” Ask your DBA to validate and compile the database.

2. Checks that details in the source database description file specification match details of the database held on the Directory.

3. Carries out the changes to the database (unloads and reloads the data). At this point any data sets, which are not being processed by this run of Fast utilities, are available for update. However, data sets that are being processed by Fast utilities are available only for read access.

4. Updates details of the database on the Directory.

NOTE

If Fast utilities fails before this point, you must recompile the database description, check and correct the data set that caused the failure, and restart program processing.

5. Invokes Batch Validate and Compile to create a new database description file that reflects the physical changes.

6. Populates index files connected to the specified database unless you specify /NOPOPULATE (no slash required under UNIX).

When the Fast utilities run finishes, the new database description is available for use. However, any user processes that were signed on to the database during the run must sign off before the new database description can be used. The database must be inactive and unloaded before subsequent user processes can access the new database description.

Fast utilities collect all informational messages issued during a run and deliver them at the end of processing. These messages do not indicate that the run failed; they are generated to make you aware of issues that may require your evaluation.

NOTE

If your database uses system logging, you must back up the database files and format a new system log before allowing any users to access the new database. For a description of a recovery point, refer to the *SUPRA Server PDM Database Administration Guide (UNIX & VMS)*, P25-2260.

Running Fast utilities when using the Multiple Physical Database facility

NOTE

Back up the SUPRA Directory and your user files before running Fast utilities.

If you use the Multiple Physical Database (MPD) facility (two databases sharing one logical database description but using different physical files to hold the data), you must run Fast utilities twice. The first Fast utilities run will update the first database description; the second Fast utilities run will update the second description to match. You should do this if you change the record length with /LOGICAL_RECORD_LENGTH (no slash required under UNIX). With other Fast utilities parameters, you may need to recompile the second connected database description. See "Recompiling all connected databases" on page 103 for more details about the effect of unloading and reloading multiple physical databases.

If the two databases both use the same data set and the same physical files, you need only run Fast utilities on one of the databases. Fast utilities modifies the physical files. However, because Fast utilities only processes one of the two databases, the data set details in the unmodified database will no longer be accurate.

To update the invalid database to reflect the changes made to the physical files, run DBA as follows:

1. Update the details on the Physical File Attributes screen for the affected data set.

2. Validate and compile the database to create a valid database description file.

3. If the old database description was loaded, use the UNLOAD operator command before you attempt to use the new database description file.

Using Fast utilities to UNLOAD or LOAD data sets (UNIX only)

In addition to being able to modify the physical attributes of data sets, you can use changedb to produce an unload (sequential) file containing all active records in a data set. You can use this unload file to load a data set. The LOAD and UNLOAD data set options of changedb provide a high-speed Unload/Reload utility, which is useful for backups or for porting data sets to other environments.

The format of the file produced by the UNLOAD data set option of changedb is a fixed record length file. The record length is 4 plus the record length defined for the data set in the database definition. Each active record is written to the file preceded by a 4-byte binary field containing the original RRN in the data set being unloaded. Each record in the unload file contains the binary image of a record in the data set, including root fields and linkpaths. The LOAD data set option expects this file format in the input file.

When you use the UNLOAD data set option, changedb performs only the unload operation. No sorting or data set modifications are performed. When you use the LOAD data set option, all data set modification options are performed.

The combination of performing an UNLOAD operation followed by a LOAD operation yields the same result as when changedb is performed without the UNLOAD/LOAD, except that an additional file is used in the UNLOAD/LOAD operation. After performing an UNLOAD operation, you can choose to delete the original data set in order to free disk space before proceeding with the LOAD operation.

Calculating disk space requirements

Fast utilities uses the DEC SORT facility under VMS and a customized sort facility under UNIX to create the sort work files. When Fast utilities modifies the data sets, it loads them from the work files into output files. Before you run Fast utilities, you must plan your disk space requirements to allow enough free space for the sort work files and the output files.

VMS

When processing very large data sets, you may not have enough disk space for both sort work files and output files. In this case, you may disregard the space requirements for the output files. If Fast utilities cannot create an output file due to lack of disk space, it pauses to allow you to delete any unnecessary storage you might have.

However, you must always ensure that you have enough space for the work files. If the DEC SORT facility cannot extend its work due to inadequate disk space, Fast utilities aborts. If this happens, restore your database from a backup copy and start again.

VMS allows up to ten work files: SORTWORK0 through SORTWORK9. By default, two work files are created in the directory assigned the logical name SYS$SCRATCH. You may specify alternative logical assignments using the VMS DEFINE command as follows:

```
$ DEFINE SORTWORKn device:[directory]filename.type
```

where *n* is a digit 0–9, inclusive.

The highest sort work file you specify determines how many sort work files Fast utilities uses. For example, if you define SORTWORK4, then five work files are used. To use any extra space that may be available and improve the performance of the sort, you can put the sort work files on different disks than the input and output files.

UNIX

Fast utilities will abort if there is not enough space for the sort work files and the output files. When increasing the size of a data set with the total_records parameter, csmchangedb will create a backup of the data set before it is modified. This backup will have a .bak extension to the data set being modified (CUST.SDB.bak). The backup file will reside in the same directory as the data set. If you have limited disk space, and you have a current backup of the data set, you may request that a backup not take place. To do this, set the logical name CSI_BAK to NO by executing the following command:

```
        csideflog -p CSI_BAK NO
```

The default for CSI_BAK is YES.

As a general rule, the free disk space needed for the sort files for primary and related data sets is approximately as follows:

Data set type	Disk space requirements for the sort file
Primary	Same size as the input file
Related	Three times the size of the output file

This section provides general information for calculating sort work and output file size. However, if you have particularly large data sets or if you have only limited disk space available, you need a more accurate method to calculate sort space. See “Primary data set processing” on page 135 and “Related data set processing” on page 135 for a more precise method of determining the amount of disk space needed for the sort work files.

Primary data set processing

You determine the size of the work files by the number of active records in the input file. Use the following rule as a guideline for calculating the disk space required.

For each active source record in a primary data set:

logical record length + 4 bytes

Related data set processing

Calculating the sort space for related data sets is a little more involved than the primary data set calculation. Fast utilities processes related data sets in four steps:

1. Creates sort work files for reordering the records.
2. Creates output related files.
3. Creates sort work files for correcting the linkpath chains.
4. Creates the output primary files.

VMS

For both steps 2 and 4, if insufficient disk space is available, you can delete the closed input files.

Because sort work files are created at two stages during processing (steps 1 and 3, previously), you can calculate the disk space requirements for the sort work files at each stage. By calculating the disk-space requirements at each stage, you do not need free disk space for all the sort work files.

VMS

Initially, you only need enough free disk space for all of the sort work files used to reorder the records. Then when Fast utilities pauses because it cannot create the first set of output files, you can delete the closed input files and continue processing. At this time, you need enough free disk space for the sort work files used to correct linkpath chains.

Use the following rules as guidelines for calculating the disk space required:

Stage 1—Free disk space needed to reorder records

Per active record:

VMS 6 bytes + primary key length + logical record length

UNIX 8 bytes + primary key length + logical record length

Stage 2—Free disk space needed to correct linkpaths

Per active record:

8 bytes + 20 bytes per nonprimary linkpath

Per nonblank linkpath (the linkpath that references the related data set from a connected primary data set):

VMS 34 bytes + twice key length

UNIX 36 bytes + twice key length

Calculating output file size

The ratio of output file size to input file size is the same as the ratio of the number of records in the output file to the number of records in the input file.

For example, if the number of records in both input and output files is the same, the ratio is 1:1. The space needed by the output file is the same as that used by the input file. If, however, the value for TOTAL_RECORDS is increased from 10,000 to 20,000, the ratio of output records to input records is now 2:1, and the output file requires twice as much disk space as the input file. This ratio may not be exact if the number of records per block has been changed, but it is useful as a general guideline.

Order of data set processing (VMS only)

If you use a change file containing modifications to both primary and related data sets, Fast utilities processes all of the primary data sets first, in the order in which they are specified; then it processes the related data sets.

NOTE

Do not use complex change files if you have limited disk space because this makes it more difficult to calculate disk space requirements.

Running Fast utilities

Run Fast utilities using the command CHANGEDB (VMS) or csmchangedb (UNIX). This section contains information about:

- **VMS** CHANGEDB.CLD file
- Fast utilities command parameters
- How to run Fast utilities using the following methods:
  - Executing using a change file
  - Executing interactively at the command line
  - Spawning as a subprocess (VMS) or executing as a background task (UNIX)
  - Executing using a command file (VMS) or a shell script (UNIX)

CHANGEDB.CLD file (VMS only)

The CHANGEDB command is described to VMS in a file called CHANGEDB.CLD. You add this file to the system command table at boot time or to your user command table at login as follows:

```
$SET COMMAND SUPRA_COMS:CHANGEDB.CLD
```

You usually carry out this procedure when you install SUPRA.

When you run Fast utilities, you specify the user name, the password, and the database name used to access the Directory; then Batch Validate and Compile by using /USERNAME, /PASSWORD and /SIGNON_DB_NAME. If you omit these parameters, CHANGEDB uses the following defaults:

```
User name: DATABASE-DESCRIPTIONS
Password:  (none)
Database:  database name on the Directory, as specified through
  the database-name parameter.
```

Fast utilities command parameters

When you enter the change database command at the command line (VMS) or the shell (UNIX), it takes a series of parameters. The parameters are classified as follows:

- **Data set parameters.** Specify the data set to be modified and the changes to be made. The parameters are:

VMS

```
/DATASET=data-set-name

/data-set-type

/TOTAL_RECORDS=new-file-size

/RECORDS_PER_BLOCK=records-per-block

/CONTROL_INTERVAL=records-in-control-interval

/LOAD_LIMIT=percentage-of-control-interval

/LOGICAL_RECORD_LENGTH=record-length

/LINK=(linkpath-name1,linkpath-name2...)

/ALLOCATION=(na1,na2...)

/FILE_SPEC=(file1,file2...)

/SHADOW=(sfile1,sfile2...)
```

UNIX

```
DATASET=data-set-name

data-set-type

TOTAL_RECORDS=new-file-size

RECORDS_PER_BLOCK=records-per-block

CONTROL_INTERVAL=records-in-control-interval

LOAD_LIMIT=percentage-of-control-interval

LOGICAL_RECORD_LENGTH=record-length

LINK=(linkpath-name1,linkpath-name2...)

ALLOCATION=na1,na2...

FILE_SPEC=(file1,file2...)

SHADOW=(sfile1,sfile2...)

LOAD

LOAD=file

UNLOAD

UNLOAD=file
```

- **Database parameters.** Supply optional database details. The parameters are:

VMS

```
database-name
/SIGNON_DB_NAME=logical-database-name
/DB_PASSWORD=database-password
/OUTPUT=target-database-description-file-spec
/NOLIST
/NOPOPULATE
```

UNIX

```
DB_NAME=database-name
SIGNON_DB_NAME=logical-database-name
DB_PASSWORD=database-password
OUTPUT=target-database-description-file-spec
NOLIST
NOPOPULATE
```

- **Directory access parameters.** Access the Directory from a user name other than the default. By default, CHANGEDB accesses the Directory through the DBA user name DATABASE-DESCRIPTIONS with no password. The parameters are:

VMS

```
/USERNAME=dba-username
/PASSWORD=dba-password
```

UNIX

```
USERNAME=dba-username
PASSWORD=dba-password
```

- **Change file parameter**. Specifies a physical file that contains a list of the changes to be made to a database (see "Executing Fast utilities using a change file" on page 156 for more information). A change file may only contain data set parameters. The parameter is:

VMS

```
/CHANGE_LIST=change-file-spec
```

UNIX

```
CHANGE_LIST=change-file-spec
```

The following provides the descriptions and syntax considerations for the Fast utilities parameters:

CHANGEDB VMS

csmchangedb UNIX

Description *Required*. The command that initiates the Fast utilities.

Format Enter exactly as shown.

/DATASET=data-set-name VMS

DATASET=*data-set-name* UNIX

Description *Required*. Specifies the name of the data set to be unloaded/reloaded.

Consideration The specified data set must exist in the database.

/data-set-type VMS

data-set-type UNIX

Description *Required*. Specifies the type of the data set to be processed.

Options /RELATED or /PRIMARY (no slash required under UNIX)

/TOTAL_RECORDS=*new-file-size* VMS

TOTAL_RECORDS=*new-file-size* UNIX

Description *Optional*. Specifies the number of records the target data set can hold.

Default Same as the input data set.

Options 2–2,147,483,647

Considerations

- For related data sets, this value is a multiple of the control interval and includes one internal record per control interval.
- For primary data sets, this value is a multiple of the block size and includes one internal record.
- You should not specify a file size that is less than the number of active records in the data set.

/RECORDS_PER_BLOCK=*records-per-block* VMS

RECORDS_PER_BLOCK=*records-per-block* UNIX

Description *Optional*. Specifies the number of records in a block.

Default Same as the input data set.

Format 1–5 numeric characters

Considerations

- The minimum value allowed is 1.
- The database validation program calculates a block size large enough to hold the specified number of records per block (in multiples of 512 bytes). The number of records per block is then rounded up to the number of records that can fit into this block size.
- Calculate the records-per-block as follows: block size divided by the record size.

/CONTROL_INTERVAL=*records-in-control-interval* VMS

CONTROL_INTERVAL=*records-in-control-interval* UNIX

Restriction Related data sets only.

Description *Optional*. Specifies the number of related records held in an internal space management unit.

Default Same as the input data set.

Format 1–10 numeric characters

Considerations

- The database validation program rounds this value up to a multiple of the records-per-block value.
- Specify a control interval between 2 and the value of TOTAL_RECORDS.

/LOAD_LIMIT=*percentage-of-control-interval* VMS

LOAD_LIMIT=*percentage-of-control-interval* UNIX

Restriction Related data sets only.

Description *Optional*. Specifies the load capacity for a control interval as a percentage.

Default Same as the input data set.

Options VMS 1–99

UNIX 0–99

Considerations

- If the percentage is exceeded, records added to existing lists of related records are added to the same control interval. To help keep lists of records in one block, the PDM starts new lists in a new control interval.
- If all control intervals are at load limit, the PDM starts new lists in any block and returns a LOAD status when you sign off from the database.
- Fast utilities ignores the load limit figure during processing, using instead a temporary load limit based on how full the target file will be. The specified load limit is used after Fast utilities processing is complete.

UNIX

If the LOAD-LIMIT is set to 0, the behavior of the related data set, space allocation mechanism is modified. The first record of a list is added in strict round-robin fashion from the first control interval to the last, then back to the first. Records added to existing chains follow the same rules defined for LOAD-LIMIT=1–99. This mechanism can be very effective in evenly loading data sets. Some data sets tend to get front-loaded when the standard LOAD-LIMIT of 1–99 is used. The LOAD-LIMIT=0 option tends to spread the chains out so that the file is more evenly distributed. A LOAD status will no longer be returned to an application on sign off when a data set is approaching the FULL condition.

/LOGICAL_RECORD_LENGTH=*record-length* VMS

LOGICAL_RECORD_LENGTH=*record-length* UNIX

Description *Optional*. Increases the logical record length in bytes.

Options Primary data sets **VMS** 21–4096

Related data sets **VMS** 41–4096

Primary data sets **UINX** 21–32,768

Related data sets **UINX** 41–32,768

Considerations

- /LOGICAL_RECORD_LENGTH (no slash required under UNIX) allows you to increase the logical record length of the data set. However, the new length will no longer match the combined length of all the data items. Before you next compile a database containing the changed data set, run SUPRA DBA to add data items so that their combined length matches the new logical record length. If the data set contains coded records, ensure that the length of the longest record matches the new logical record length. Once the combined length of all the data items in the data set matches the logical record length specified during the Fast utilities run, validate and compile the database ready for use.
- Application programs can continue to use the data set, provided you do not validate and compile it.

/LINK=(*linkpath-name1,linkpath-name2,...*) VMS

LINK=(*linkpath-name1,linkpath-name2,...*) UNIX

Restriction Related data sets only.

Description *Optional*. Specifies the name of the primary linkpath to be optimized by Fast utilities.

Format 8 alphanumeric characters

Considerations

- ♦ Specify the primary linkpath first in the list to improve database access to consecutive records on this linkpath.
- ♦ Any linkpaths which are not specified are added to the list in the order in which they appear in the related data set.
- ♦ To specify a series of linkpath names, separate each with a comma and enclose the string in parentheses.

/ALLOCATION=(*na1,na2,...*) VMS

ALLOCATION=*na1,na2,...* UNIX

Description *Optional*. Specifies the relative size of each physical disk file for the specified data set.

Default Same as the input data set.

Format 1–4 numeric characters

Considerations

- You can specify a maximum of 4 allocation values.
- The number of allocation values should equal the number of output file specifications.
- You may want to specify more physical files than were specified in the source database description. To do this, specify more allocation values than there are input file specifications, and ensure that you specify one output file specification for each allocation value.
- To specify a series of allocation values, separate each value with a comma and enclose the string in parentheses.
- Do not specify an allocation value if you have only one output file specification.

/FILE_SPEC=(*file1,file2,...*) VMS

FILE_SPEC=(*file1,file2,...*) UNIX

Description *Optional*. Specifies the physical disk file(s) assigned to hold all or part of the data set.

Default Same as the input data set.

Format Up to 44 alphanumeric characters. Must be a valid VMS file specification or UNIX pathname.

Considerations

- You can specify a maximum of 4 file specifications.
- If you omit an output file specification, the file specifications in the source database are used.
- You can use existing file names by specifying a null string, for example, /FILE_SPEC=(new1.dat,"",new3.dat) (no slash required under UNIX).
- The number of output file specifications must equal the number of allocation values. To specify more than one output file, separate the file names with commas and enclose the string in parentheses.
- To specify only one output file, omit the parentheses and commas.

/SHADOW=(*sfile1,sfile2,...*) VMS

SHADOW=(*sfile1,sfile2,...*) UNIX

Description *Optional*. Specifies the physical disk files used to shadow the output files.

Default Same as the input data set.

Format Up to 44 alphanumeric characters. Must be a valid VMS file specification or UNIX pathname.

Considerations

- You can specify a maximum of 4 shadow file specifications.
- You can use existing file names by specifying a null string (/SHADOW=(sfile1.dat,sfile2.dat,"")) (no slash required under UNIX).
- Store the shadow files on a different device from the main files.
- Specify one shadow file for each output file. To specify more than one shadow file, separate the file names with commas and enclose the string in parentheses.
- To specify only one shadow file, omit the parentheses and commas.

LOAD[=*file*] UNIX

Description *Optional*. Indicates that changedb will use an unload (sequential) file as input rather than a data set.

Default If you do not use the LOAD data set option, the data set is used as input and the data set is either UNLOADed or modified according to the data set options specified on the command line or in the change_list file.

Format The file name, if specified, can be up to 64 characters long and can include a valid UNIX pathname.

Considerations

- The record length of the unload file must be 4 plus the record length of the data set.
- The unload file may be created on a different platform or database. It is important to format the records with the proper data during the unload process. Primary record linkpath data should be set to NULL (binary 0). These data will be rebuilt by the LOAD process. The related record linkpath data must be valid. The previous and next pointers must contain either NULL (binary 0) or valid 4-byte binary RRN values. The RRN values must point to valid RRNs in the original data set. The first 4 bytes of each unload record must contain the original RRN of the record. This RRN value must match the previous and next linkpath values which point to it.
- If you do not specify a file name, the default file name of the unload file will be *data_set_name*.unload. If you are loading the CUST data set, changedb will attempt to open the file CUST.unload in the current directory.
- All data set modification options are available when you use the LOAD data set option.
- The file specification can be a tape device such as /dev/rmt/0m. In this case, the tape must have been created by the csmchangedb utility using the UNLOAD data set option, or by a program that has written records to the tape that are 4 plus the record size defined in the database definition for this data set. Blocks may be written to the tape in even multiples of records up to 65K in length.
- When LOADing a related data set, changedb can also perform complete reorganization on connected primary data sets.

UNLOAD[=*file*] UNIX

Description *Optional*. Indicates that changedb will unload the data set to an unload (sequential) file.

Default If you do not use the UNLOAD data set option, changedb modifies the physical attributes of the data set according to the data set options specified on the command line or in the change_list file.

Format The file name, if specified, can be up to 64 characters long and can include a valid UNIX pathname.

Considerations

- All active records in the data set will be unloaded to the unload file. The unload file will contain fixed length records. The record size will be 4 plus the record size defined for the data set in the database definition. The first 4 bytes will contain the binary RRN value of the original record in the data set. No data conversion is done on the data contained in the records. Each record in the unload file will contain the binary image of a record in the data set, including root fields and linkpaths.

- If you do not specify a file name, the default file name of the unload file will be *data_set_name*.unload. If you are unloading the CUST data set, changedb will attempt to open the file CUST.unload in the current directory.

- No other data set operations are performed on the data set when you use the UNLOAD data set option.

- The file specification can be a tape device such as /dev/rmt/0m. In this case, all active records in the data set will be written to the tape. Each record on the tape will be 4 plus the record size in the data set. This is the format required by the LOAD data set option. Records will be written to the tape in blocks containing an even multiple of records up to 65K in length.

database-name VMS

DB_NAME=***database-name*** UNIX

Description *Required*. Specifies the name of the compiled database description on the Directory.

Format 6 alphanumeric characters

Considerations

- If you do not specify a SIGNON_DB_NAME, Fast utilities finds the file to be read by appending the file extension ".MOD" (VMS) or .mod (UNIX) to the specified database name and translating logical assignments.
- The database must exist on the Directory.
- If the database is active, then an error status is returned and Fast utilities aborts.

/USERNAME=***dba-username*** VMS

USERNAME=***dba-username*** UNIX

Description *Conditional*. Required in UNIX environments; optional in VMS environments. Specifies the DBA user name that is to access the Directory.

Default DATABASE-DESCRIPTIONS

Format 1–30 alphanumeric characters. Must be a valid DBA user name.

Considerations

- If the DBA user name has a matching password, the user name must match the password you specify for the /PASSWORD parameter (UNIX does not require the slash).
- The DBA user name must have a privilege of DATABASE-ADMINISTRATOR or above.

/PASSWORD=*dba-password* VMS

PASSWORD=*dba-password* UNIX

Description *Conditional*. Required under UNIX; optional under VMS. Specifies the password for the user name that is to access the Directory.

Default No password

Format 8 alphanumeric characters. Must correspond to the DBA user name.

Considerations

- If the DBA user name has a password, the password must match the user name you specified in the /USERNAME parameter (no slash required under UNIX). If you enter an invalid password, Fast utilities aborts and returns you to the command level (VMS) or the shell (UNIX).

- The password is displayed on the screen as you type. To avoid displaying the password, omit /PASSWORD. Fast utilities then prompts you to enter the DBA user password, and does not display what you enter.

/SIGNON_DB_NAME=*logical-database-name* VMS

SIGNON_DB_NAME=*logical-database-name* UNIX

Description *Conditional*. Required under UNIX; optional under VMS. Specifies the name used by application programs to sign on to the database.

Default Database name specified through the database-name parameter.

Format 6 alphanumeric characters

Considerations

- CHANGEDB finds the file to be read by appending the file extension .MOD (VMS) or .mod (UNIX) to the database name and translating any logical assignments.

- The database name in the compiled database description may differ from the database name on the Directory to allow for multiple physical databases.

- Only use this qualifier if you specified a “logical database description name” when you compiled the database.

/DB_PASSWORD=*database-password* VMS

DB_PASSWORD=*database-password* UNIX

Description *Required.* Only if the database is protected by a password; specifies the password assigned to the database.

Format 1–6 alphanumeric characters

Considerations

- If the database description has a password, the password you enter must match the database description password. If you enter an invalid password, Fast utilities aborts and returns you to the command level (VMS) or the shell (UNIX).
- The password is displayed on the screen as you type. To avoid displaying the password, omit /DB_PASSWORD (no slash required under UNIX). Fast utilities prompts you to enter the database password and does not display what you enter.

/OUTPUT=*target-database-description-file-spec* VMS

OUTPUT=*target-database-description-file-spec* UNIX

Description *Optional*. Identifies the file specification of the target database description.

Default The same name as the input-compiled database description file.

Format VMS Valid file specification

UNIX Valid pathname

Considerations

- You only need to use this parameter if the target database description file name is to be different from the source database description file name.
- Using this parameter does not affect the /database-name parameter and the /SIGNON_DB_NAME parameter.

/NOLIST VMS

NOLIST UNIX

Description *Optional*. Inhibits the production of a database print file after compilation.

Default /LIST (no slash required under UNIX)

Consideration Omit this parameter to produce a database listing called database-name.LIS (VMS) or database-name.lis (UNIX).

/CHANGE_LIST=*change-file-spec* VMS

CHANGE_LIST=*change-file-spec* UNIX

Description *Optional*. Specifies a file containing a list of modifications.

Format VMS Valid file specification

UNIX Valid pathname

Considerations

- This parameter allows you to modify more than one data set in a single run.
- When you use /CHANGE_LIST (no slash required under UNIX), you may enter no data set parameters; the information is held in the change-file-spec.
- When you use a change file, the database-name parameter is required. However, the Directory access parameters /USERNAME and /PASSWORD, and the database parameters /SIGNON_DB_NAME, /DB_PASSWORD and /OUTPUT (no slash required under UNIX) are optional.

/NOPOPULATE VMS

NOPOPULATE UNIX

Description *Optional*. Inhibits the automatic population of any connected indices at the end of the Fast utilities run.

Default /POPULATE (no slash required under UNIX)

Considerations

- ♦ Omit /NOPOPULATE (no slash required under UNIX) to populate all indices after the Fast utilities run.
- ♦ This parameter is required when using the /OUTPUT (no slash required under UNIX) parameter and your database uses indices.

Executing Fast utilities using a change file

VMS

You can change multiple data sets in a single run by specifying a change file on the command line. To create and use a change file, follow these two steps:

1. Create a change file using a standard text editor as shown in the following example (lines with an exclamation mark in the first character space are comments):

```
!Change file CHANGES.DAT for database CUSTDB

/DATASET=RELD/RELATED

/DATASET=PRIM/PRIMARY/TOTAL_RECORDS=100000

/DATASET=RELA/RELATED/ALLOCATION=(2,1,1) -

/FILE_SPEC=(DB:RELA01.DAT,DB:RELA02.DAT,",")/LINK=PRIMLKRE
```

2. Invoke Fast utilities with the change file in the format:

```
/CHANGE_LIST=change-file database-name
```

   For example, to run Fast utilities with the example change file above, enter:

```
$ CHANGEDB/CHANGE_LIST=CHANGES.DAT CUSTDB
```

You may include only data set parameters in a change file; you cannot include the Directory access parameters /USERNAME and /PASSWORD, and the database parameters SIGNON_DB_NAME, DB_PASSWORD and /OUTPUT. Specify these parameters either on the command line or in the command file, if you use one.

See “Calculating disk space requirements” on page 133 for information about how to establish the disk space you need for each data set modified during a change file run. If you do not have enough space for the creation of all output files, the Fast utilities pause for you to delete closed input files.

However, you must have enough free disk space for the work files to allow all of the data sets to be changed. If Fast utilities should abort during a change file run, you may have difficulty establishing which data sets have been processed and which have not.

UNIX

You can change multiple files in a single run by listing the Fast utilities parameters in a change file. To create and use a change file, follow these two steps:

1. Create a change file using a standard text editor as shown in the following example (lines with a # in the first character are comments):

```
#change file changes.dat for database custdb

DATASET=test1 PRIMARY TOTAL_RECORDS=10000

DATASET=test2 RELATED TOTAL_RECORS=20000 LOAD_LIMIT=90

DATASET=test3 RELATED TOTAL_RECORDS=10000
```

2. Initiate Fast utilities with the change file in the following format:

```
csmchangedb CHANGE_LIST=/path/file-name
            DB_NAME=database-name

    USERNAME=user-name
```

For example, to run Fast utilities with the above change file, enter the following:

```
csmchangedb CHANGE_LIST=/usr/changes.dat DB_NAME=custdb

    USERNAME=user-name
```

You may include only data set parameters in a change file; you cannot include the Directory access parameters USERNAME and PASSWORD, and the database parameters SIGNON_DB_NAME, DB_PASSWORD and OUTPUT. Specify these parameters either at the shell or in the command file, if you use one.

If you do not have enough free disk space for the sort work files and the output files, Fast utilities will abort. See “Calculating disk space requirements” on page 133 for calculating the amount of disk space you need for each data set modified during a change file run.

Executing Fast utilities interactively

To execute Fast utilities interactively, enter the command with all of the parameters and database-name parameter at the command line. The minimum form of the command is:

VMS

```
$CHANGEDB/DATASET=data-set-name/data-set-type database-name
```

UNIX

```
csmchangedb DATASET=data-set-name data-set-type
          DB_NAME=database-name
```

For example, enter the following to reorganize the data set RELD along the first linkpath in the base record:

VMS

```
$ CHANGEDB /DATASET=RELD/RELATED CUSTDB
```

UNIX

```
csmchangedb DATASET=reld related DB_NAME=custdb
```

The following sections show some of the ways you can use Fast utilities to modify the physical characteristics of a database.

Using Fast utilities to alter file size

The following example changes the size of the data set PRIM:

VMS

```
$ CHANGEDB DATASET=PRIM/PRIMARY/TOTAL_RECORDS=100000 CUSTDB
```

UNIX

```
csmchangedb DATASET=PRIM PRIMARY TOTAL_RECORDS=100000
          USERNAME=kirk DB_NAME=custdb
```

Using Fast utilities to reorganize a related data set

The following examples modify the database CUSTDB. The allocation for the three physical disk files has been set to a ratio of 2:1:1. This means that the first file holds twice as many records as either of the other two.

VMS

/FILE_SPEC specifies the physical disk files to be used. The last file specification is the same as the original output file, indicated with the null string ",".

/LINK specifies the primary linkpath as PRIMLKRE. This is the linkpath along which optimization occurs.

```
$ CHANGEDB/DATASET=RELA/RELATED/ALLOCATION=(2,1,1) -

_$ /FILE_SPEC=(DB:RELA01.DAT,DB:RELA02.DAT,",")
            /LINK=PRIMLKRE CUSTDB
```

UNIX

FILE_SPEC specifies the physical disk files to be used.

LINK specifies the primary linkpath as PRIMLKRE. This is the linkpath along which optimization occurs.

```
csmchangedb DATASET=RELA RELATED ALLOCATION=(2,1,1) \

FILE_SPEC
  =/directory/path/RELA01.dat,/directory/path/RELA02.dat

LINK=PRIMLKRE DB_NAME=custdb
```

Using Fast utilities to correct linkpath errors

Warning: Fast utilities can correct some ICHN linkpath errors; however, Cincom recommends that you correct all physical errors (and therefore logical errors) by running system log recovery on your backups, or by reapplying all transactions since the last backup.

You can use Fast utilities to correct some ICHN linkpath errors. However, Fast utilities *cannot* correct a primary record pointing to nonexistent related records, or related records that have no parent primary record. If you have either of these linkpath errors, you can use the DBA unload/reload utilities to correct them.

Fast utilities only corrects errors along the first linkpath specified in the /LINK (no slash required under UNIX) list. Therefore, if the inconsistent linkpath is the primary linkpath, run Fast utilities once and specify the primary linkpath first in the /LINK (no slash required under UNIX) list. However, if the secondary linkpath is inconsistent, you must run Fast utilities twice on the same data set as follows:

- **Initial run.** Specify the inconsistent secondary linkpath in the first position in the /LINK list (no slash required under UNIX) (because Fast utilities only corrects errors along the first linkpath specified)

- **Second run.** Specify the primary linkpath in the first position in the /LINK (no slash required under UNIX) list

Spawning Fast utilities as a subprocess (VMS) or executing it as a background task (UNIX)

VMS

To spawn a Fast utilities subprocess, type SPAWN CHANGEDB at the command line together with a list of parameters and the database-name parameter. The following shows the format:

```
$SPAWN CHANGEDB/DATASET=data-set-name/data-set-type/...
```

For example:

```
$ SPAWN CHANGEDB/DATASET=PRIM/PRIMARY/TOTAL_RECORDS=100000 CUSTDB
```

You can also spawn a subprocess using a change file. The format for doing so follows:

```
$SPAWN CHANGEDB/CHANGE_LIST=change-file database-name
```

For example, the following specifies an output file with the SPAWN command. /OUTPUT is associated with the SPAWN command and is unrelated to the CHANGEDB OUTPUT parameter. It allows you to continue to use the terminal while the Fast utilities execute.

```
$ SPAWN/NOWAIT/OUTPUT=CUSTDB.STATS -

_$ CHANGEDB/CHANGE_LIST=CHANGES.DAT CUSTDB
```

UNIX

To execute Fast utilities as a background process, enter the csmchangedb command with qualifiers:

```
nohup csmchangedb dataset=data-set-name data-set-type

➢  output-filename 2>>output-file-name
```

For example, the following will run Fast utilities as a background process and place the report in the current directory in a file named changedb.out:

```
nohup csmchangedb dataset=prim primary \

> changedb.out 2>> changedb.out
```

Executing a Fast utilities command file (VMS) or script (UNIX)

VMS

The following is an example command file, DO_CHANGEDB.COM, to expand the size of the related data set ADDR to 250,000 records. Because Fast utilities exit with a status indicating whether the process succeeded, you can include a status check within the command file. A status of ERROR indicates you must solve the problem before resubmitting the run. A status of WARNING indicates that either the database or the data set is still active. In this case, the run will probably succeed if you resubmit it at a later time.

```
$IF P1 .EQS. "" THEN EXIT
$ON WARNING       THEN GOTO RESUBMIT
$ON ERROR         THEN GOTO FAIL
$
$CHANGEDB/DATASET='P1'/TOTAL_RECORDS=250000 CUSTDB
$WRITE SYS$OUTPUT "CHANGEDB SUCCEEDED"
$EXIT
$
$RESUBMIT: !resubmit the same job after a delay of 1 hour
$SUBMIT/PARAM='P1'/AFTER="+1:00:00" DO_CHANGEDB
$EXIT
$
$FAIL:     !a severe error occurred, so exit
$WRITE SYS$OUTPUT "CHANGEDB FAILED"
$EXIT
```

UNIX

The following is an example script csmchangedb.com, to change the number of records to 250,000:

```
#changedb script
csmchangedb DATASET=addr RELATED TOTAL RECORDS=250000 \
DB_NAME=custdb USERNAME=DATABASE-DESCRIPTIONS \
>changed.out 2>>changedb.out
```

Executing a Fast utilities command file (VMS) or script (UNIX) online

To execute the command file from the command line, use the following format:

VMS

```
$@command-file.COM
```

UNIX

```
shell-script.com
```

The following example shows how to run Fast utilities using the example command files in “Executing a Fast utilities command file (VMS) or script (UNIX)” on page 162:

VMS

```
$ @DO_CHANGEDB.COM ADDR
```

UNIX

```
csmchangedb.com
```

You may also specify a change file in the command file.

Submitting a Fast utilities command file (VMS) or script (UNIX) to batch

You may submit a Fast utilities command file at the command level (VMS) or shell (UNIX) using the following format:

VMS

```
SUBMIT command-file.COM
```

UNIX

```
command-file.com &
```

For example, to submit the command file described in “Executing a Fast utilities command file (VMS) or script (UNIX)” on page 162, enter:

VMS

```
$ SUBMIT /PARAM=ADDR DO_CHANGEDB.COM
```

UNIX

```
command-file.com &
```

4

Using SUPRA PDM Batch Directory Maintenance (DIRM) (VMS)

Batch Directory Maintenance (DIRM) is an alternative to the DBA utility for the VMS environment. DIRM uses a batch script file with a specific format (described in "Batch DIRM statements" on page 173) to export and import metadata for databases. The utility itself is interactive and menu-driven.

NOTE

DIRM supersedes the Database Transfer utility previously available in VAX environments.

NOTE

If you have multiple groupwide or systemwide PDM environments, you are limited to using DIRM in one environment, and in only one process, at a time to keep the data from getting crossed up during Batch Directory Maintenance work.

If you are familiar with Cincom's MANTIS application development environment, you may use the MANTIS batch language commands to execute this utility from a submitted command procedure.

There are two logicals that you will need to use to run from a submitted command procedure if you are not familiar with MANTIS. More infomation on these can be found with the example at the end of the chapter.

NOTE

MANTIS is a Cincom application development environment closely integrated with SUPRA Server. MANTIS is not required to operate DIRM. If you wish to learn more about MANTIS, contact your Cincom account representative.

NOTE

Warning: As always, when making a change to your database definition, create a recovery point prior to any major Batch DIRM run (as described in the *SUPRA Server PDM System Administration Guide (VMS)*, P25-0130).

Setting up Batch DIRM

The files required for using DIRM are delivered in the directory identified by the logical SUPRA_REPORT. The command procedure SUPRA_COMS:SUPRA_SYMBOL.COM defines the symbol DIRM, which will be used to activate the Batch Directory Maintenance utility.

To use Batch DIRM, perform the following steps:

1. Log in to an OpenVMS account that has access to your SUPRA Directory database, SUPRAD. SUPRAD is the repository for the metadata for your database. (For details on setting up a PDM environment, refer to the *SUPRA Server PDM System Administration Guide (VMS)*, P25-0130. For details on setting up a database, refer to the *SUPRA Server PDM Database Administration Guide (UNIX & VMS)*, P25-2260.)
2. If you will be using the LOAD function of Batch DIRM, then define the logical DIRM_INPUT to identify the file that contains the Batch DIRM input statements.
3. If you operate with the Cincom MANTIS application development environment, ensure that you have no existing MANTIS Class named DIRM. (MANTIS is not required to operate DIRM.)
4. Establish WRITE access to:
   ```
   SUPRA_REPORT:MANTIS_DIRM_27.DATA_*
   ```
5. Enter $DIRM to use the symbol DIRM (defined in the command procedure SUPRA_COMS:SUPRA_SYMBOL.COM). This command executes the command procedure SUPRA_COMS:RUNDIRM.COM. The following description of RUNDIRM.COM includes MANTIS-specific information for those familiar with that product. The RUNDIRM.COM script:
   - Defines the logical DIRM_VIEWS to point to the directory containing the SUPRAD.GBL file (*not* to the file itself).
   - Uses a private MANTIS_CLASS called DIRM.
   - Defines the logical MANTIS_FILE to point to:
     `SUPRA_REPORT:MANTIS_DIRM_27.DATA_`*`xxx`*

     where *xxx* = VAX or AXP, depending on your environment.
   - Runs the special, execute-only MANTIS image provided in the directory identified by the logical SUPRA_REPORTS, bypassing the DIRM sign-on screen.
   - Signs on using the user name DIRM for normal Batch DIRM operation. The password is blank by default.
6. A menu is presented. The options are described in “Operating DIRM” on page 168.

Batch DIRM restrictions

The following restrictions currently exist when using Batch DIRM:

- The SUPRAD database definition cannot be modified.
- Program-related entities are not supported.
- The DATABASE-STATUS is set to NEEDS-VALIDATION only if a database name was found somewhere in the batch input. When Batch DIRM is used to modify an entity without specifying the database description of which it is a part, the DATABASE-STATUS is not changed. After modifications to your database description have been made, you must use the DBA utility to validate and compile your database description(s).

NOTE

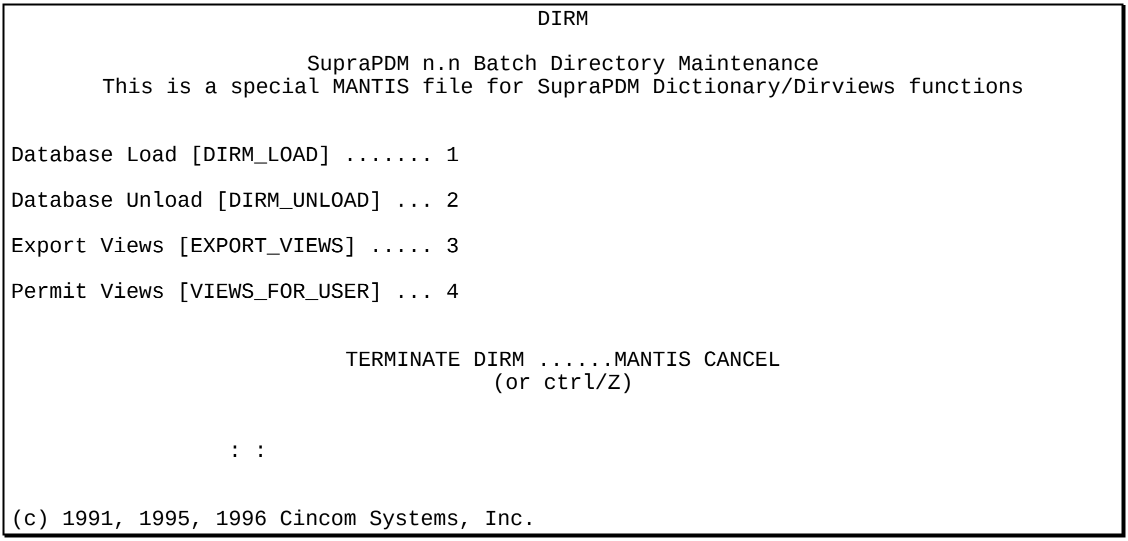
Some entities, like data sets, can be associated with multiple database descriptions.

- Shadow-file-specs and files-specs-2–4 are not supported. For a discussion of data set file specifications, refer to the *SUPRA Server PDM Database Administration Guide (UNIX & VMS)*, P25-2260.
- The MAX-TASKS parameter of your database description will be made equal to MAX-UPDATE-TASKS when the latter field is modified. For a discussion of these parameters, refer to the *SUPRA Server PDM Database Administration Guide (UNIX & VMS)*, P25-2260.
- Batch DIRM will not perform cascade deletes like DBA utility. For example, if you want to delete a data set, delete all of the data items first, which in turn requires the logical data items, indices, and so on, to be deleted first. This is done for security reasons because Batch DIRM cannot ask you for confirmation of these high-impact deletes.
- If you want to add comments to objects added with Batch DIRM, you must use Batch DIRM's COMMENT MOD function to create at least the first line of comment. CSIDBA will not allow you to add comments to an object created with Batch DIRM if no comments exist. It is advisable to use Batch DIRM to maintain comments on objects created by Batch DIRM. See "The Batch DIRM input file" on page 172 and "Batch DIRM statements" on page 173 for a detailed description of the DIRM_INPUT file.
- DIRM will keep dictionary entities in alphabetical sequence where appropriate, but it will not change existing sequences.

Operating DIRM

Warning: Create a recovery point (back up) of your SUPRA Directory database (SUPRAD) files (located using the logical definition CSI_DIRDB) prior to making any modifications. Refer to the *SUPRA Server PDM System Administration Guide (VMS)*, P25-0130, for a detailed description of SUPRAD and the logical CSI_DIRDB. Refer to the *SUPRA Server PDM Database Administration Guide (UNIX & VMS)*, P25-2260, for a detailed description of a recovery point.

Below is the Batch Directory Maintenance menu:

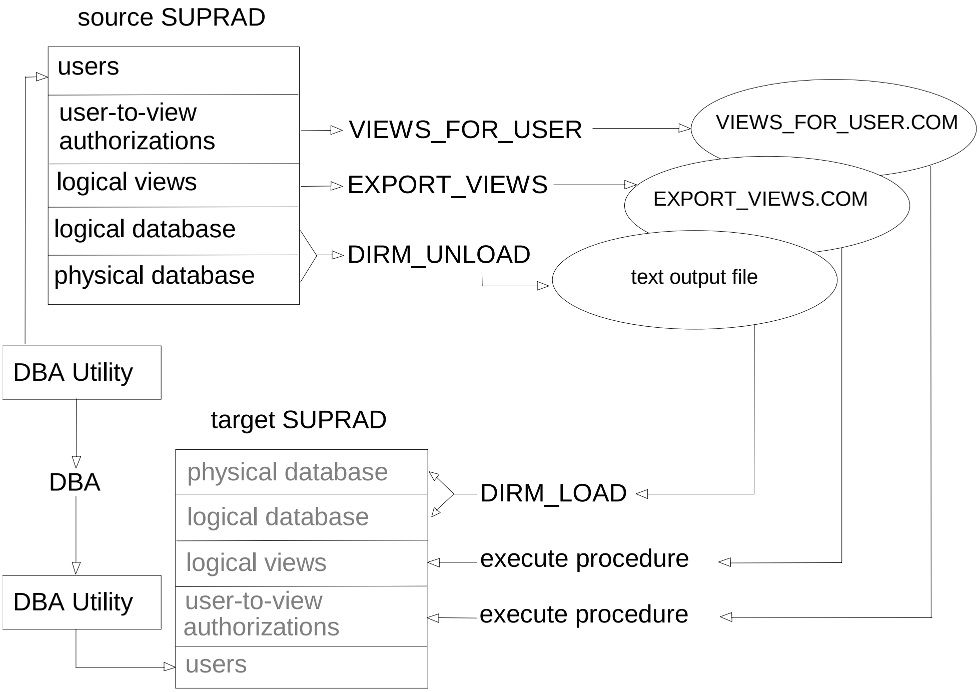


```
                                         DIRM

                       SupraPDM n.n Batch Directory Maintenance
       This is a special MANTIS file for SupraPDM Dictionary/Dirviews functions


Database Load [DIRM_LOAD] ....... 1

Database Unload [DIRM_UNLOAD] ... 2

Export Views [EXPORT_VIEWS] ..... 3

Permit Views [VIEWS_FOR_USER] ... 4


                          TERMINATE DIRM ......MANTIS CANCEL
                                      (or ctrl/Z)


                 : :


(c) 1991, 1995, 1996 Cincom Systems, Inc.
```

The Batch Directory Maintenance menu provides the following functions:

- **DIRM_LOAD**. Allows you to process existing Batch DIRM input files.
- **DIRM_UNLOAD**. Allows you to unload the metadata for an existing database description or data set from the Directory into a text file containing Batch DIRM input statements. This allows you to apply your dictionary updates online using the DBA utility, followed by a DIRM_UNLOAD to have a Batch DIRM copy of your updates for later use.
- **EXPORT_VIEWS**. Allows you to create a DBAID DCL script file, which is used to recreate RDM view definitions.
- **VIEWS_FOR_USER**. Allows you to create a DBAID DCL script file, which is used to recreate RDM view permissions.

Performing a database transfer. The following figure shows how to transfer a database:



To transfer a database, perform the following steps:

1. Ensure your VMS process environment (logicals and symbols) is set up to use the existing (source) SUPRA Directory database (SUPRAD) holding the definition of your database.

2. Use the symbol DIRM to run Batch Directory Maintenance.

3. At the DIRM menu, select option 2, Database Unload (DIRM_UNLOAD). This option will unload the definition of your database description (except for any logical views and user-to-view permissions) into a file. Take note of the displayed name of the output file created by the UNLOAD, which will be used later in the LOAD phase.

4. When DIRM_UNLOAD is complete, exit Batch Directory Maintenance. (DIRM options 3 and 4 use a different global view file than options 1 and 2.)

5. Use the symbol DIRM to run Batch Directory Maintenance.

6. At the DIRM menu, select option 3, Export Views (EXPORT_VIEWS). This option will create a batch DBAID command procedure with the name EXPORT_VIEWS.COM, which will contain the view definitions for this database description.

7. At the DIRM menu, select option 4, Permit Views (VIEWS_FOR_USER). This option will create a batch DBAID command procedure with the name VIEWS_FOR_USER.COM. This file will contain the user to view permissions as defined for this database.

8. Write down the users and passwords you will be adding to the new SUPRA Directory database. Include all users who will have access to any of your database entities or programs.

9. The UNLOAD phase is complete. Exit Batch Directory Maintenance.

10. Either sign on to an OpenVMS account with access to the new (target) SUPRA Directory database or create the new database environment.

11. Gain access to the file created in step 3.

12. Define the logical DIRM_INPUT to point to this file.

13. (*Optional*) Using an editor, modify the DIRM_INPUT file to match your requirements for this new system. See "Batch DIRM statements" on page 173 for a description of the DIRM input file statements.

14. Gain read access to the batch DBAID command procedures EXPORT_VIEWS.COM and VIEWS_FOR_USER.COM (from steps 6 and 7).

15. If you made any changes to the DIRM_INPUT file that will affect the logical views exported from the source database, use an editor to make the needed corresponding changes to EXPORT_VIEWS.COM or VIEWS_FOR_USER.COM.

16. Use the symbol DIRM to run Batch Directory Maintenance.

17. At the DIRM menu, select option 1, Database Load (DIRM_LOAD). This will load the definition of your database description (except for any logical views and user-to-view permissions) from the file identified by the logical DIRM_INPUT. The DIRM_LOAD option sets the following two DCL symbols upon exit to inform you on the result of the run. You should test for these symbols:

DCL symbol	**Value**	**Return status**
DIRM_EXIT	OK	Normal run without errors
	ERRORS	Normal run with errors
	ABORTED	Run was aborted
DIRM_ERRORS		Contains error count or -1 if run was aborted

18. Exit Batch Directory Maintenance.

19. Using the DBA utility, add the users and their passwords noted in step 8.

20. At the command level, execute the batch DBAID command procedure EXPORT_VIEWS.COM. This will load your logical views.

21. Execute the batch DBAID command procedure VIEWS_FOR_USER.COM. This will load the user-to-view permissions for your logical views.

22. Use the DBA utility to finish making the new database description available for use. This will include validating and compiling the database description and formatting files. For a discussion of the DBA utility, refer to the *SUPRA Server PDM Database Administration Guide (UNIX & VMS)*, P25-2260.

The Batch DIRM input file

The Batch DIRM input file (identified by the logical DIRM_INPUT) is a text file containing Batch DIRM statements. These statements describe the database description or entities you choose. There are two ways to create a Batch DIRM input file:

- DIRM option 2, DIRM_UNLOAD, will export entity definitions from an existing database description into a text file which you can alter with a text editor
- You can use a text editor to manually create your Batch DIRM input file

Once DIRM_INPUT contains the information you want, you can load it into the SUPRA Directory database (SUPRAD) using DIRM option 1, DIRM_LOAD. The Batch DIRM input file must conform to the following conventions:

- An exclamation point in the first column indicates a comment line.
- A colon in the first column indicates a comment line that is copied to MANTIS_OUTPUT (generally SYS$OUTPUT) when processed.
- Each DIRM statement must start with a category code, followed by an action name, followed by one or more parameters separated by commas or spaces. Statement syntax is discussed in detail in “Batch DIRM statements” on page 173.
- Each statement must end with a period.
- A statement may use more than one line.
- Except for the category code and action name (which must be the first two verbs, in that order), all parameters are position-independent.
- You may use spaces or tabs to increase the readability of the input file.
- All lowercase letters are converted to uppercase, except for words enclosed in double quotes (which can be lowercase or mixed-case).
- The last statement in the Batch DIRM input file must be:

  ```
  END.
  ```

Batch DIRM statements

A Batch DIRM statement consists of a database entity category (BUFF for buffer), followed by a command stating how that entity will be affected (ADD), followed by one or more parameters describing which part(s) of the entity the command will act upon (NAME, TYPE).

The syntax for a Batch DIRM statement is:

***category-name action-name*, {*parameter-name*=*value*, ...}**
[*parameter-name*=*value*, ...]{.}

Example

```
DITM ADD, DSET=PART, NAME=DESC, AFTER=CTRL, TYPE=C, SIZE=20.
```

The following categories are available with Batch DIRM:

Category	Description
BUFF	Buffer
COMMENT	Replace comments for the previous object
DBDESC	Database description
DITM	Physical data item
DOMN	Domain
DSET	Data set
END.	End of statement stream
FSPC	File spec; modify physical file attributes
INDX	Index
LGDI	Logical data item
OPTION	Options modification for this DIRM run
RECD	Record code
RMSK	RMS key
SKEY	Secondary key
SLOG	System log
TLOG	Task log
USER	SUPRA user name
VTBL	Validation table

The following sections provide the categories, actions, parameters, and considerations for Batch DIRM statements. For more specific information on the entities described by the Batch DIRM parameters, refer to the *SUPRA Server PDM Database Administration Guide (UNIX & VMS)*, P25-2260.

BUFF category

The BUFF category includes the following actions:

- ADDing a buffer (see “Adding a buffer” below)
- MODifying a buffer (see “Modifying a buffer” on page 176)
- DELeting a buffer (see “Deleting a buffer” on page 177)
- CONnecting a buffer (see “Connecting a buffer” on page 178)
- DISconnecting a buffer (see “Disconnecting a buffer” on page 179)

For command syntax, refer to the appropriate section.

Adding a buffer

To add a buffer, use the following syntax:

BUFF ADD NAME=*buffer-name* DBDESC=*database-name*

TYPE=*data-set-type* [COPIES=*no.-of-buffer-copies*]

[DSET=*data-set-name*]

NAME=*buffer-name*

Description *Required*. Specifies the name of the buffer being added.

Format 4 characters

DBDESC=*database-name*

Description *Required*. Specifies the name of the database to which this buffer is being added.

Format 6 characters

TYPE=*data-set-type*

Description *Required*. Specifies the type of data set which will use this buffer.

Options R Related

P Primary

COPIES=*number-of-buffer-copies*

Description *Optional*. Specifies the number of copies of the buffer.

Default 5

Options 1–32,000

DSET=*data-set-name*

Description *Optional*. Specifies the data set to which the buffer is being added.

Format 4 characters

Modifying a buffer

To modify a buffer, use the following syntax:

BUFF MOD NAME=*buffer-name* DBDESC=*database-name*

COPIES=*no.-of-buffer-copies*

NAME=*buffer-name*

Description *Required*. Specifies the name of the buffer being modified.

Format 4 characters

DBDESC=*database-name*

Description *Required*. Specifies the name of the database whose buffer is being modified.

Format 6 characters

COPIES=*number-of-buffer-copies*

Description *Required*. Specifies the number of copies of this buffer that can be used by the selected database description.

Options 1–32000

Deleting a buffer

To delete a buffer, use the following syntax:

BUFF DEL NAME=*buffer-name* DBDESC=*dbase-name*

[DSET=*data-set-name*]

NAME=*buffer-name*

Description *Required*. Specifies the name of the buffer being deleted.

Format 4 characters

DBDESC=*database-name*

Description *Required*. Specifies the name of the database whose buffer is being deleted.

Format 6 characters

DSET=*data-set-name*

Description *Optional*. Specifies the data set to disconnect the buffer from before the delete is attempted.

Format 4 characters

Connecting a buffer

To connect a buffer, use the following syntax:

BUFF CON NAME=*buffer-name* DBDESC=*database-name*

DSET=*data-set-name*

NAME=*buffer-name*

Description *Required*. Specifies the name of the buffer being connected to the data set (specified by DSET= below).

Format 4 characters

DBDESC=*database-name*

Description *Required*. Specifies the name of the database to which this buffer and data set belong.

Format 6 characters

DSET=*data-set-name*

Description *Required*. Specifies the data set to connect to this buffer.

Format 4 characters

Disconnecting a buffer

To disconnect a buffer, use the following syntax:

```
BUFF DIS NAME=buffer-name DBDESC=database-name
  DSET=data-set-name
```

NAME=*buffer-name*

Description *Required*. Specifies the name of the buffer to disconnect from the data set.

Format 4 characters

DBDESC=*database-name*

Description *Required*. Specifies the name of the database whose data set and buffer are being disconnected.

Format 6 characters

DSET=*data-set-name*

Description *Required*. Specifies the data set being disconnected from the buffer.

Format 4 characters

COMMENT category

The COMMENT category allows you to replace comments for the previous object. To perform this action, use the following syntax:

COMMENT MOD

COMMENT works on the object modified in the previous command. If you want to replace the comments for an object, you must have a MOD statement before the COMMENT statement:

```
DITM MOD, NAME=CTRL, DSET=PART.
COMMENT MOD.
```

COMMENT MOD deletes all of the existing comments for the object and then adds all of the following input lines until a line is seen starting with:

```
END-COMMENT.
```

DBDESC category

The DBDESC category includes the following actions:

- ADDing a database description (see “Adding a database description” on page 181)
- MODifying a database description (see “Modifying a database description” on page 183)
- DELeting a database description (see “Deleting a database description” on page 185)

For command syntax, refer to the appropriate section.

Adding a database description

To add a database description, use the following syntax:

DBDESC ADD NAME=*database-name* HELD=*nnn* TASKS=*nnnn*
[SHADOW=*option*] [SINGLE=*option*] [ACCESS=*access-method*]
[GBLSECT=*option*] [CALLING=*option*] [CLUSTER=*option*]

NAME=*database-name*

Description *Required*. Specifies the name of the database being added.

Format 6 characters

HELD=*nnn*

Description *Required*. Specifies the maximum number of records that a single update user can hold.

Options 0–999

TASKS=*nnnn*

Description *Required*. Specifies the maximum number of tasks that may be signed on concurrently in read-only or update modes.

Options 1–1275

SHADOW=*option*

Description *Optional*. Specifies how shadow files are used if a read or write fails on a data set or log (I/O error).

Default N

Options

C	Continue processing the file that has not failed.
D	Deny access to the entire database.
F	Deny access to the data set or log in error.
N	Do not perform shadow recording regardless of whether a shadow file is defined.
O	Notify and prompt the console operator to handle the error.

SINGLE=*option*

Description *Optional*. Indicates whether one or multiple users can use the database concurrently.

Default N

Options

Y	One user
N	Multiple users

ACCESS=*access-method*

Description *Optional*. Specifies how the PDM will access the database files.

Default Q

Options

Q	QIO for local files, RMS for remote files
R	RMS for all files

GBLSECT=*option*

Description *Optional*. Defines the global sections used to access the database description module.

Default G

Options

G	Group
S	System

CALLING=*option*

Description *Optional*. Specifies the method used to pass parameters to the PDM.

Default R

Options

R	Reference
D	Reference/Descriptor

CLUSTER=*option*

Description *Optional*. Specifies whether to run this copy of SUPRA in a clustered environment.

Default L

Options

L	Local
C	Cluster/network

Modifying a database description

To modify a database description, use the following syntax:

DBDESC MOD NAME=*database-name* [HELD=*nnn*] [TASKS=*nnnn*]

[SHADOW=*option*] [SINGLE=*option*] [ACCESS=*access-method*]

[GBLSECT=*option*] [CALLING=*option*] [CLUSTER=*option*]

NAME=*database-name*

Description *Required*. Specifies the name of the database to be modified.

Format 6 characters

HELD=*nnn*

Description *Optional*. Specifies the maximum number of records that can be held by a single update user.

Options 0–999

TASKS=*nnnn*

Description *Optional*. Specifies the maximum number of tasks that may be signed on concurrently in read-only or update modes.

Options 1–1275

SHADOW=*option*

Description *Optional*. Specifies how shadow files are used if a read or write fails on a data set or log (I/O error).

Default N

Options

C	Continue processing the file that has not failed.
D	Deny access to the entire database.
F	Deny access to the data set or log in error.
N	Do not perform shadow recording regardless of whether a shadow file is defined.
O	Notify and prompt the console operator to handle the error.

SINGLE=*option*

Description *Optional*. Indicates whether one or multiple users can use the database concurrently.

Default N

Options

Y	One user
N	Multiple users

ACCESS=*access-method*

Description *Optional*. Specifies how the PDM will access the database files.

Default Q

Options

Q	QIO for local files, RMS for remote files
R	RMS for all files

GBLSECT=*option*

Description *Optional*. Defines the global sections used to access the database description module.

Default G

Options

G	Group
S	System

CALLING=*option*

Description *Optional*. Specifies the method used to pass parameters to the PDM.

Default R

Options

R	Reference
D	Reference/Descriptor

CLUSTER=*option*

Description *Optional*. Specifies whether to run this copy of SUPRA in a clustered environment.

Default L

Options

L	Local
C	Cluster/network

Deleting a database description

To delete a database description, use the following syntax:

DBDESC DEL NAME=*database-name*

NAME=*database-name*

Description *Required*. Specifies the name of the database to be deleted.

Format 6 characters

DITM category

The DITM category includes the following actions:

- ADDing a physical data item (see “Adding a physical data item” on page 187)
- MODifying a physical data item (see “Modifying a physical data item” on page 191)
- DELeting a physical data item (see “Deleting a physical data item” on page 194)

For command syntax, refer to the appropriate section.

Adding a physical data item

Warning: Contrary to CSIDBA, you must define the ROOT and CTRL fields explicitly.

To add a physical data item, use the following syntax:

DITM ADD NAME=*data-item name* DSET=*data-set-name*

SIZE=*data-item-size* [AFTER=*option*] [TYPE=*data-type*]

SIGN=*option* [DECIMALS= *number-of-decimal-places*]

REFER=*data-item-name* OWNER=*data-set-name*

[PARENT=*data-item-name*] CODE=*record-code*

NAME=*data-item name*

Description *Required*. Specifies the name of the data item to be added.

Format 4 characters

DSET=*data-set-name*

Description *Required*. Specifies the name of the data set to which the data item being added will belong.

Format 4 characters

SIZE=*data-item-size*

Description *Required*. Specifies the size of the data item.

Options 1–4088

AFTER=*option*

Description *Optional*. Specifies the data item after which this data item is to be added. AFTER may not be used together with PARENT. The new data-item will be created at the same sublevel as the AFTER-item.

Default If omitted, the data item is inserted at the BOTtom of the data item list for top-level fields, and at the TOP of a sublevel if PARENT is specified.

Options 4-character data item name

8-character data item name (if the AFTER name is LK*xx* in a related file)

TOP

BOT

TYPE=*data-type*

Description *Optional*. Specifies the kind of data to be stored in this data item.

Default C

Options B Binary

C Character

F Floating point

K Kanji

L Leading separate numeric

N Numeric trailing overpunch

P Packed decimal

Z Zoned trailing numeric

SIGN=*option*

Description *Conditional*. Required for Binary, Numeric, or Packed decimal data items. Specifies whether a numeric field is signed or unsigned.

Default S

Options S Signed

U Unsigned

Consideration Packed decimal and floating point data items must be signed. This option is ignored for character data items.

DECIMALS= *number-of-decimal-places*

Description *Optional*. The number of decimal places for numeric data items.

Default 0

Format 1–5 numeric characters

REFER=*data-item-name*

Description *Conditional*. Specifies the name of the referback data item. Required for LK*xx* data items in related files.

Format 4 characters

OWNER=*data-set-name*

Description *Conditional*. Specifies the data set that owns the linkpath to which the referback field will be associated. Required for LK*xx* data items in related files.

Format 4 characters

PARENT=*data-item-name*

Description *Optional*. Specifies the data item to which the new data item is to be a sub-data-item.

Format 4 characters

Considerations

- The parent must already exist.
- The new data item is put before any existing items for the sublevel.
- PARENT cannot be used together with AFTER.

CODE=*record-code*

Description *Conditional*. Required if the data item is to be part of a coded record.

Format 2 characters

Consideration The record code must already exist.

Modifying a physical data item

To modify a physical data item, use the following syntax:

DITM MOD NAME=*data-item-name* DSET=*data-set-name*

[SIZE=*data-item-size*] [TYPE=*data-type*] SIGN=*option*

[DECIMALS=*number-of-decimal-places*]

REFER=*data-item-name-of-refer-back*

OWNER=*data-set-name* CODE=*record-code*

NAME=*data-item-name*

Description *Required*. Specifies the name of the data item to be modified.

Format 4 characters

DSET=*data-set-name*

Description *Required*. Specifies the data set to which the data item belongs.

Format 4 characters

SIZE=*data-item-size*

Description *Optional*. Specifies the length of the physical data item.

Options 1–4088

TYPE=*data-type*

Description *Optional*. Specifies the kind of data to be stored in this data item.

Default C

Options

B	Binary
C	Character
K	Kanji
L	Leading separate numeric
F	Floating point
N	Numeric trailing overpunch
P	Packed decimal

SIGN=*option*

Description *Conditional*. Required for Binary, Numeric, or Packed decimal data items. Specifies whether a numeric field is signed or unsigned.

Default S

Options

S	Signed
U	Unsigned

Considerations

- ♦ Packed decimal and floating point data items must be signed.
- ♦ This option is ignored for character data items.

DECIMALS=*number-of-decimal-places*

Description *Optional*. The number of decimal places for numeric data items.

Default 0

Format 1–5 numeric characters

REFER=*data-item-name-of-refer-back*

Description *Conditional*. Specifies the name of the referback data item. Required for LK*xx* data items in related files.

Format 4 characters

OWNER=*data-set-name*

Description *Conditional*. Specifies the data set owning the linkpath. Required for LK*xx* items in related files.

Format 4 characters

Consideration The OWNER cannot be changed but must be specified to identify the linkpath.

CODE=*record-code*

Description *Conditional*. Required if the data-item to be modified is in a coded record.

Format 2 characters

Deleting a physical data item

Warning: Contrary to CSIDBA, you must define the ROOT and CTRL fields explicitly for primary data sets. Therefore, these fields must also be deleted.

To delete a physical data item, use the following syntax:

DITM DEL NAME=*data-item-name* DSET=*data-set-name*

OWNER=*data-set-name* CODE=*record-code* CODE=*record-code*

NAME=*data-item-name*

Description *Required*. Specifies the name of the data item to be deleted.

Format 4 characters

DSET=*data-set-name*

Description *Required*. Specifies the data set to which this data item belongs.

Format 4 characters

OWNER=*data-set-name*

Description *Conditional*. Specifies the data set owning the linkpath. Required for LK*xx* items in related files.

Format 4 characters

Consideration The OWNER must be specified to identify the linkpath.

CODE=*record-code*

Description *Conditional*. Required if the data item to be deleted is in a coded record.

Format 2 characters

Considerations

- A delete of a data item will disconnect it from any domain.
- If the data item is connected to a logical name, a secondary key, or an RMS key, the delete will fail; you must disconnect first.

DOMN category

The DOMN category includes the following actions:

- ADDing a domain (see “Adding a domain” on page 196)
- MODifying a domain (see “Modifying a domain” on page 201)
- DELeting a domain (see “Deleting a domain” on page 205)
- CONnecting a domain (see “Connecting a domain” on page 206)
- DISconnecting a domain (see “Disconnecting a domain” on page 207)

For command syntax, refer to the appropriate section.

Adding a domain

To add a domain, use the following syntax:

DOMN ADD NAME=*domain-name* SIZE=*domain-size* [TYPE=*data-type*]

SIGN=*option* [DECIMALS=*number-of-decimal-places*]

[NALLOW=*option*] NVALUE=*null-value* [DEFAULT=*default-value*]

[RVAL=*option*] VALTYPE=*option* MIN=*minimum-value*

MAX=*maximum-value* EXIT=*exit-name*

TABLE=*validation-table-name*

NAME=*domain-name*

Description *Required*. Specifies the name of the domain to be added.

Format 1–30 characters

SIZE=*domain-size*

Description *Required*. Specifies the size of the domain to be added.

Options 1–4088K

Consideration Refer to DOMAIN-LENGTH in the *SUPRA Server PDM Database Administration Guide (UNIX & VMS)*, P25-2260.

TYPE=*data-type*

Description *Optional*. Specifies the data type of the domain to be added.

Default C

Options B Binary

C Character

F Floating point

K Kanji

L Leading separate numeric

N Numeric trailing overpunch

P Packed decimal

Z Zoned trailing numeric

Consideration The domain FUNCTION will be STRING if TYPE=C; it will be NUMBER for all other TYPEs.

SIGN=*option*

Description *Conditional*. Required for Binary, Numeric, or Packed decimal data items. Specifies whether a numeric field is signed or unsigned.

Default S

Options S Signed

U Unsigned

Considerations

- Packed decimal and floating point data items must be signed.
- This option is ignored for character data items.

DECIMALS=*number-of-decimal-places*

Description *Optional*. The number of decimal places for numeric data items.

Default 0

Format 1–5 numeric characters

NALLOW=*option*

Description *Optional*. Specifies whether the data item may contain a null value.

Default N

Options Y Null values allowed

N No null values allowed

Consideration Refer to DOMAIN-NULLS-ALLOWED in the *SUPRA Server PDM Database Administration Guide (UNIX & VMS)*, P25-2260.

NVALUE=*null-value*

Description *Required.* If NALLOWED is Y, specifies the character(s) or values to be used to indicate a null value.

Default Spaces

Format Must be a valid value for the data items associated with this domain.

DEFAULT=*default-value*

Description *Optional*. Specifies the default value.

Default Spaces

Format Must be a valid value for the data items associated with this domain.

Consideration This value must be enclosed in double quotes (") if the type is Character and the value starts with a nonalphabetic or contains special characters.

RVAL=*option*

Description *Optional*. Specifies whether RDM will perform validation during GET.

Default N

Options

Y Retrieval validation

N No retrieval validation

Consideration If RVAL=Y, you must further specify the type of validation through the next four fields.

VALTYPE=*option*

Description *Required* when RVAL=Y.

Default N

Options

N None

R Range

T Table

E Exit

MIN=*minimum-value*

Description *Conditional*. Required if VALTYPE=R (range). Specifies the minimum value in the range.

Consideration This value must be enclosed in double quotes (") if the type is Character and the value starts with a nonalphabetic or contains special characters.

MAX=*maximum-value*

Description *Conditional*. Required if VALTYPE=R (range). Specifies the maximum value in the range.

Considerations

- This value must be greater than the MIN.
- This value must be enclosed in double quotes (") if the type is Character and the value starts with a nonalphabetic or contains special characters

EXIT=*exit-name*

Description *Conditional*. Required if VALTYPE=E (exit). Specifies the name of the entry point into the validation exit module (not the VMS file name).

Format 1–8 characters

TABLE=*validation-table-name*

Description *Conditional*. Required if VALTYPE=T (table). Specifies the name of the validation table to be used by this domain.

Format 1–30 characters

Modifying a domain

To modify a domain, use the following syntax:

DOMN MOD NAME=*domain-name* [SIZE=*domain-size*]

[TYPE=*data-type*] SIGN=*option*

[DECIMALS=*number-of-decimal-places*]

[NULLS=*option*] NVALUE=*null-value* [DEFAULT=*default-value*]

[RVAL=*option*] VALTYPE=*option* MIN=*minimum-value*

MAX=*maximum-value* EXIT=*exit-name*

TABLE=*validation-table-name*

NAME=*domain-name*

Description *Required*. Specifies the name of the domain to be modified.

Format 1–30 characters

SIZE=*domain-size*

Description *Optional*. Specifies the size of the domain.

Options 1–4096

TYPE=*data-type*

Description *Optional*. Specifies the data type of the domain to be modified.

Default C

Options

B Binary

C Character

F Floating point

K Kanji

L Leading separate numeric

N Numeric trailing overpunch

P Packed decimal

Z Zoned trailing numeric

Consideration The domain FUNCTION will be STRING if TYPE=C; it will be NUMBER for all other TYPEs.

SIGN=*option*

Description *Conditional*. Required for Binary, Numeric, or Packed decimal data items. Specifies whether a numeric field is signed or unsigned.

Default S

Options

S Signed

U Unsigned

Consideration Packed decimal and floating point data items must be signed. This option is ignored for character data items.

DECIMALS=*number-of-decimal-places*

Description *Optional*. The number of decimal places for numeric data items.

Default 0

Format 1–5 numeric characters

NULLS=*option*

Description *Optional*. Specifies whether the data item may contain a null value.

Default N

Options
Y Null values allowed

N No null values allowed

Consideration Refer to DOMAIN-NULLS-ALLOWED in the *SUPRA Server PDM Database Administration Guide (UNIX & VMS)*, P25-2260.

NVALUE=*null-value*

Description *Required* if NALLOWED is Y. Specifies the character(s) or values to be used to indicate a null value.

Default Spaces

Format Must be a valid value for the data items associated with this domain.

DEFAULT=*default-value*

Description *Optional*. Specifies the default value.

Default Spaces

Format Must be a valid value for the data items associated with this domain.

Consideration This value must be enclosed in double quotes (") if the type is Character and the value starts with a nonalphabetic or contains special characters.

RVAL=*option*

Description *Optional*. Specifies whether RDM will perform validation during GET.

Default N

Options
Y Retrieval validation

N No retrieval validation

Consideration If RVAL=Y, you must further specify the type of validation through the next four fields.

VALTYPE=*option*

Description *Conditional*. Required when RVAL=Y.

Default N

Options

N	None
R	Range
T	Table
E	Exit

MIN=*minimum-value*

Description *Conditional*. Required if VALTYPE=R (range). Specifies the minimum value in the range.

Consideration This value must be enclosed in double quotes (") if the type is Character and the value starts with a nonalphabetic or contains special characters.

MAX=*maximum-value*

Description *Conditional*. Required if VALTYPE=R (range). Specifies the maximum value in the range.

Considerations

- This value must be greater than the MIN.
- This value must be enclosed in double quotes (") if the type is Character and the value starts with a nonalphabetic or contains special characters.

EXIT=*exit-name*

Description *Conditional*. Required if VALTYPE=E (exit). Specifies the name of the entry point into the validation exit module (not the VMS file name).

Format 1–8 characters

TABLE=*validation-table-name*

Description *Conditional*. Required if VALTYPE=T (table). Specifies the name of the validation table to be used by this domain.

Format 1–30 characters

Deleting a domain

To delete a domain, use the following syntax:

DOMN DEL NAME=*domain-name*

NAME=*domain-name*

Description *Required*. Specifies the name of the domain to be deleted.

Format 1–30 characters

Connecting a domain

To connect a domain, use the following syntax:

DOMN CON NAME=*domain-name* DSET=*data-set-name-of-data-item*

DITM=*data-item-name*

NAME=*domain-name*

Description *Required*. Specifies the name of the domain to which the specified data item will be connected.

Format 1–30 characters

DSET=*data-set-name-of-data-item*

Description *Required*. Specifies the name of the data set whose data item is being connected to this domain.

Format 4 characters

DITM=*data-item-name*

Description *Required*. Specifies the name of the data item to connect to this domain.

Format 4 characters

Disconnecting a domain

To disconnect a domain, use the following syntax:

DOMN DIS NAME=*domain-name* DSET=*data-set-name-of-data-item*

DITM=*data-item-name-to-disconnect-from*

NAME=*domain-name*

Description *Required*. Specifies the name of the domain from which the data item is being disconnected.

Format 1–30 characters

DSET=*data-set-name-of-data-item*

Description *Required*. Specifies the name of the data set whose data item is being disconnected from the domain.

Format 4 characters

DITM=*data-item-name-to-disconnect-from*

Description *Required*. Specifies the name of the data item to disconnect from the domain.

Format 4 characters

DSET category

The DSET category includes the following actions:

- ADDing a data set (see "Adding a data set" below)
- MODifying a data set (see "Modifying comments for a data set" on page 209)
- DELeting a data set (see "Deleting a data set" on page 209)
- CONnecting a data set (see "Connecting a data set" on page 210)
- DISconnecting a data set (see "Disconnecting a data set" on page 212)

For command syntax, refer to the appropriate section.

Adding a data set

To add a data set, use the following syntax:

DSET ADD NAME=*data-set-name* TYPE=*data-type*

NAME=*data-set-name*

Description *Required*. Specifies the name of the data set to be added.

Format 4 characters

TYPE=*data-type*

Description *Required*. Specifies the data set type.

Options

P Primary

R Related

X RMS data sets

Modifying comments for a data set

To modify comments for a data set, use the following syntax:

DSET MOD NAME=*data-set-name*

NAME=*data-set-name*

Description *Required*. Specifies the name of the data set whose comments are being modified.

Format 4 characters

Consideration Nothing in the data set can be modified. Use FSPC MOD to modify physical file specs. Use DSET MOD only to modify comments for the DSET.

Deleting a data set

To delete a data set, use the following syntax:

DSET DEL NAME=*data-set-name* [DBDESC=*database-name*]

NAME=*data-set-name*

Description *Required*. Specifies the name of the data set to be deleted.

Format 4 characters

DBDESC=*database-name*

Description *Optional*. Disconnects the data set from the database specified before attempting to delete the data set.

Format 6 characters

Consideration DSET DEL requires that all DITMs are deleted first.

Connecting a data set

To connect a data set, use the following syntax:

```
DSET CON NAME=data-set-name DBDESC=database-name
  FSPEC=OpenVMS-file-spec RPB=records-per-block
  RECS=total-records [CI=records-per-control-interval]
  [LL=load-limit] [MODE=access-mode] [BUCKET=bucket-size]
```

NAME=*data-set-name*

Description *Required*. Specifies the name of the data set to be connected to the specified database description.

Format 4 characters

DBDESC=*database-name*

Description *Required*. Specifies the name of the database to which the data set is being connected.

Format 6 characters

Consideration Within data-set-type, DIRM maintains the data sets in alphabetical order.

FSPEC=*OpenVMS-file-spec*

Description *Required*. Specifies the physical file name for the data set.

Format 1–44 characters. If the file-spec contains special characters, it must be enclosed in double quotes (").

RPB=*records-per-block*

Description *Conditional*. Required for primary and related data sets. Specifies the number of records in a logical block.

Options 1–32K

RECS=*total-records*

Description *Conditional*. Required for primary and related data sets. Specifies the total number of logical records the data set can hold.

Options 2–99,999,999

CI=*records-per-control-interval*

Description *Optional*. Related files only. Specifies the number of related records held in a control interval (PDM internal space management unit).

Default RPB value

Options 2–32K

LL=*load-limit*

Description *Optional*. Related files only. Specifies the load capacity for a control interval as a percentage.

Default 80%

Options 1–100%

MODE=*access-mode*

Restriction Do not enter this option for RMS data sets; it is not allowed.

Description *Optional*. Specifies the access mode for the data set.

Default U

Options U Update

R Read-only

BUCKET=*bucket-size*

Description *Optional*. RMS data sets only. Specifies the number of disk pages in each RMS bucket (block).

Default 0

Options 0–63

Disconnecting a data set

To disconnect a data set, use the following syntax:

DSET DIS NAME=*data-set-name* DBDESC=*database-name*

NAME=*data-set-name*

Description *Required*. Specifies the name of the data set to be disconnected from the specified database description.

Format 4 characters

DBDESC=*database-name*

Description *Required*. Specifies the database from which to disconnect the data set.

Format 6 characters

END. category

The END. category denotes the end of the command stream. There are no action parameters. To end the command stream, enter the following:

END.

FSPC category

The FSPC category allows you to modify physical-file attributes. To modify physical-file attributes, use the following syntax:

FSPC MOD DBDESC=*database-name* DSET=*data-set-name*

[FSPEC=*OpenVMS-file-spec*] [RPB=*records-per-block*]

[RECS=*total-records*] [CI=*records-per-control-interval*]

[LL=*load-limit*] [MODE=*access-mode*] [BUCKET=*bucket-size*]

DBDESC=*database-name*

Description *Required*. Specifies the database whose data set's physical-file attributes are being changed.

Format 6 characters

DSET=*data-set-name*

Description *Required*. Specifies the name of the data set whose physical-file specifications are being changed.

Format 4 characters

FSPEC=*OpenVMS-file-spec*

Description *Optional*. Specifies the OpenVMS file specification for the specified data set and database.

Format 1–44 characters. If the file-spec contains special characters, it must be enclosed in double quotes (").

RPB=*records-per-block*

Description *Optional*. Specifies the number of logical records in a logical block.

Options 1–32K

Consideration Not allowed for RMS data sets.

RECS=*total-records*

Description *Optional*. Specifies the total number of logical records the data set can hold.

Options 2–99,999,999

Consideration Not allowed for RMS data sets.

CI=*records-per-control-interval*

Description *Optional*. Related files only. Specifies the number of related records held in a control interval (PDM internal space management unit).

Default RPB value

Options 2–99,999,999

LL=*load-limit*

Description *Optional*. Related files only. Specifies the load capacity for a control interval as a percentage.

Default 80%

Options 1–100%

MODE=*access-mode*

Description *Optional*. Specifies the access mode for the data set.

Default U

Options U Update

R Read-only

Consideration Not allowed for RMS data sets.

BUCKET=*bucket-size*

Description *Optional*. RMS data sets only. Specifies the number of disk pages in each RMS bucket (block).

Default 0

Options 0–63

INDX category

The INDX category includes the following actions:

- ADDing an index (see "Adding an index" on page below)
- MODifying an index (see "Modifying an index" on page 217)
- DELeting an index (see "Deleting an index" on page 219)
- CONnecting an index (see "Connecting an index" on page 220)
- DISconnecting an index (see "Disconnecting an index" on page 222)

For command syntax, refer to the appropriate section.

Adding an index

To add an index, use the following syntax:

INDX ADD NAME=*index name* DSET=*data-set-name*

NAME=index name

Description *Required*. Specifies the name of the index to be added.

Format IX*xx*

DSET=data-set-name

Description *Required*. Specifies the data set name to which the index belongs.

Format 4-character data-set name

Consideration Use INDX CON to connect the index to a database.

Modifying an index

To modify an index, use the following syntax:

INDX MOD NAME=*index-name* DSET=*data-set-name*

DBDESC=*database-name* [FSPEC=*OpenVMS-file-spec*]

[CORRUPT=*option*] [NULLS=*option*] [VERIFY=*option*]

NAME=*index-name*

Description	*Required*. Specifies the name of the index to be modified.
Format	IX*xx*

DSET=*data-set-name*

Description	*Required*. Specifies the data set name to which the index belongs.
Format	4 characters

DBDESC=*database-name*

Description	*Required*. Specifies the name of the database to which changes refer.
Format	6 characters

FSPEC=*OpenVMS-file-spec*

Description	*Optional*. Specifies the OpenVMS physical file specification for the index.
Format	1–44 characters. If the file-spec contains special characters, it must be enclosed in double quotes (").

CORRUPT=*option*

Description *Optional*. Specifies the action the PDM should take if it detects a corrupt index file.

Default None

Options O Operator

C Continue

NULLS=*option*

Description *Optional*. Specifies where to sort null values in the collating sequence.

Default None

Options H High

L Low

VERIFY=*option*

Description *Optional*. Specifies whether the PDM is to check for a corrupt index when reading an index file.

Default None

Options Y Yes

N No

Deleting an index

To delete an index, use the following syntax:

INDX DEL NAME=*index-name* DSET=*data-set-name*

[DBDESC=*database-name*]

NAME=*index-name*

Description *Required*. Specifies the name of the index to be deleted.

Format IX*xx*

DSET=*data-set-name*

Description *Required*. Specifies the name of the data set to which the index belongs.

Format 4 characters

DBDESC=*database-name*

Description *Optional*. Specifies the name of the database to disconnect from before attempting the index deletion.

Format 6 characters

Considerations

- All related secondary keys must be deleted first.
- The index must not be connected to any (other) database.

Connecting an index

To connect an index, use the following syntax:

INDX CON NAME=*index-name* DSET=*data-set-name*

DBDESC=*database-name* FSPEC=*OpenVMS-file-spec*

[CORRUPT=*option*] [NULLS=*option*] [VERIFY=*option*]

NAME=*index-name*

Description *Required*. Specifies the name of the index to be connected to the specified data set and database description.

Format IX*xx*

DSET=*data-set-name*

Description *Required*. Specifies the name of the data set to which the index belongs.

Format 4 characters

DBDESC=*database-name*

Description *Required*. Specifies the name of the database to which the index can connect.

Format 6 characters

FSPEC=*OpenVMS-file-spec*

Description *Required*. The physical file specification for this index.

Format 1–44 characters. If the file-spec contains special characters, it must be enclosed in double quotes (").

CORRUPT=*option*

Description *Optional*. Specifies the action the PDM should take if it detects a corrupt index file.

Default None

Options

O Operator

C Continue

NULLS=*option*

Description *Optional*. Specifies where to sort null values in the collating sequence.

Default None

Options

H High

L Low

VERIFY=*option*

Description *Optional*. Specifies whether the PDM is to check for a corrupt index when reading an index file.

Default None

Options

Y Yes

N No

Disconnecting an index

To disconnect an index, use the following syntax:

INDX DIS NAME=*index-name* DSET=*data-set-name*

DBDESC=*database-name*

NAME=*index-name*

Description *Required*. Specifies the name of the index to be modified.

Format IX*xx*

DSET=*data-set-name*

Description *Required*. Specifies the name of the data set to which the index belongs.

Format 4 characters

DBDESC=*database-name*

Description *Required*. Specifies the name of the database from which to disconnect the index.

Format 6 characters

LGDI category

The LGDI category includes the following actions:

- ADDing a logical data item (see “Adding a logical data item” on page 224)
- MODifying a logical data item (see “Modifying comments for a logical data item” on page 225)
- DELeting a logical data item (see “Deleting a logical data item” on page 225)

For command syntax, refer to the appropriate section.

Adding a logical data item

To add a logical data item, use the following syntax:

LGDI ADD NAME=*logical-data-item-name* DSET=*data-set-name*

DITM=*data-item-name*

NAME=*logical-data-item-name*

Description *Required*. Specifies the name of the logical data item to be added.

Format 1–26 characters

Consideration You can change the maximum length of a logical data item by using OPTION MOD, LGDISIZE=*nn*.

DSET=*data-set-name*

Description *Required*. Specifies the name of the data set to which the data item belongs.

Format 4 characters

DITM=*data-item-name*

Description *Required*. Specifies the name of the physical data item associated with the logical name.

Format 4 characters

Modifying comments for a logical data item

To modify comments for a logical data item, use the following syntax:

LGDI MOD NAME=*logical-data-item-name*

NAME=*logical-data-item-name*

Description *Required*. Specifies the logical name to be modified.

Format 1–26 characters

Considerations

- You cannot modify the logical data item; use LGDI MOD only to modify comments for the logical data item.
- You can change the maximum length of a logical data item by using OPTION MOD, LGDISIZE=*nn*.

Deleting a logical data item

To delete a logical data item, use the following syntax:

LGDI DEL NAME=*logical-data-item-name*

NAME=*logical-data-item-name*

Description *Required*. Specifies the logical name to be deleted.

Format 1–26 characters

Consideration The logical data item must not be connected to any view.

OPTION category

The OPTION category allows you to modify certain characteristics of the current DIRM run. To modify the current DIRM run, use the following syntax:

OPTION MOD [ECHO=*option*] [GOTO=*label-name*]

[LGDISIZE=*max.-length-of-an-LGDI*] [MAXERR=*nnnn*] [STATS]

ECHO=*option*

Description *Optional*. Controls echoing of input lines to SYS$OUTPUT.

Default N

Options N No echoing

A All lines echoed

C All lines except comment lines (!) echoed

Consideration Comment lines are always echoed

GOTO=*label-name*

Description *Optional*. Specifies the name of the label to which control will be passed.

Format 1–44 characters

Consideration DIRM skips input lines until a line is found in the format :*label-name*: (starting and ending with a colon), where *label-name* identifies a label that is below this statement (the label to which the GOTO refers must be physically after the GOTO statement in the DIRM_INPUT file).

LGDISIZE=*max.-length-of-an-LGDI*

Description *Optional*. Specifies the maximum length of an LGDI.

Default 26

Options 10–30

Consideration Cincom recommends a maximum value of 26 characters.

MAXERR=*nnnn*

Description *Optional*. Specifies the maximum number of DIRM errors allowed.

Default 10

Options 0–9999

Consideration The exact number of errors after which DIRM aborts may be somewhat greater.

STATS

Description *Optional*. Prints run statistics (CPU, Direct I/O, etc.).

Consideration Statistics will be printed at the end of the DIRM run.

RECD category

The RECD category includes the following actions:

- ADDing a record code (see “Adding a record code” on page 228)
- MODifying a record code (see “Modifying comments for a record code” on page 229)
- DELeting a record code (see “Deleting a record code” on page 230)

For command syntax, refer to the appropriate section.

Adding a record code

To add a record code, use the following syntax:

RECD ADD NAME=*record-code* DSET=*data-set-name*

NAME=*record-code*

Description *Required*. Specifies the record code to be added to the specified data set.

Format 2 characters

DSET=*data-set-name*

Description *Required*. Specifies the name of the data set to which the record code will be added.

Format 4 characters

General consideration

This action adds a record code to a related data set. You add a related data set with DSET ADD.

Modifying comments for a record code

To modify comments for a record code, use the following syntax:

RECD MOD NAME=*record-code* DSET=*data-set-name*

NAME=*record-code*

Description *Required*. Specifies the record code for which comments are to be modified.

Format 2 characters

DSET=*data-set-name*

Description *Required*. Specifies the name of the data set whose record code comments are being modified.

Format 4 characters

Consideration You cannot modify the record code. Use RECD MOD only to modify comments for the record code.

Deleting a record code

To delete a record code, use the following syntax:

RECD DEL NAME=*record-code* DSET=*data-set-name*

NAME=*record-code*

Description *Required*. Specifies the record code to be deleted.

Format 2 characters

DSET=*data-set-name*

Description *Required*. Specifies the name of the data set whose record code is being deleted.

Format 4 characters

General consideration

This action deletes a record code from a related data set. You can delete a related data set with DSET DEL.

RMSK category

The RMSK category includes the following actions:

- ADDing an RMS key (see “Adding an RMS key” below)
- MODifying an RMS key (see “Modifying an RMS key” on page 234)
- DELeting an RMS key (see “Deleting an RMS key” on page 235)

For command syntax, refer to the appropriate section.

Adding an RMS key

To add an RMS key, use the following syntax:

```
RMSK ADD NAME=RMS-key-name DSET=RMS-data-set-name
  KEYNO=RMS-key-number DITM=first-data-item-in-key
  [LENGTH=RMS-key-length] [UNIQUE=option] [CHANGES=option]
```

NAME=*RMS-key-name*

Description *Required*. Specifies the name of the RMS key to be added to the specified RMS data set.

Format 1–8 characters

DSET=*RMS-data-set-name*

Description *Required*. Specifies the name of the RMS data set to which the RMS key is being added.

Format 4 characters

KEYNO=*RMS-key-number*

Description *Required*. Specifies the RMS key as either the primary or alternate key of reference to the RMS data set.

DITM=*first-data-item-in-key*

Description *Required*. Specifies the name of the first data item in the key.

Format 4 characters

LENGTH=*RMS-key-length*

Description *Optional*. Specifies the size of the RMS record key.

Default Size of DITM

Format 1–255

UNIQUE=*option*

Description	*Optional*. Specifies whether duplicate values are allowed for this key.
Default	Y KEYNO=0, else N
Options	Y Yes
	N No

CHANGES=*option*

Description	*Optional*. Specifies whether modifications to an existing key field are allowed.
Default	N KEYNO=0, else Y
Options	Y Yes
	N No

Modifying an RMS key

To modify an RMS key, use the following syntax:

```
RMSK MOD NAME=RMS-key-name DSET=RMS-data-set-name
  [LENGTH=RMS-key-length] [UNIQUE=option] [CHANGES=option]
```

NAME=*RMS-key-name*

Description *Required*. Specifies the name of the RMS key to be modified.

Format 1–8 characters

DSET=*RMS-data-set-name*

Description *Required*. Specifies the name of the RMS data set whose RMS key is being modified.

Format 4 characters

LENGTH=*RMS-key-length*

Description *Optional*. Specifies the size of the RMS record key.

Format 1–255

UNIQUE=*option*

Description *Optional*. Specifies whether duplicate values are allowed for this key.

Default Y KEYNO=0, else N

Options Y Yes

N No

CHANGES=*option*

Description *Optional*. Specifies whether modifications to an existing key field are allowed.

Default N KEYNO=0, else Y

Options Y Yes

N No

Deleting an RMS key

To delete an RMS key, use the following syntax:

RMSK DEL NAME=*RMS-key-name* DSET=*RMS-data-set-name*

NAME=*RMS-key-name*

Description *Required*. Specifies the name of the RMS key to be deleted.

Format 1–8 characters

DSET=*RMS-data-set-name*

Description *Required*. Specifies the name of the RMS data set whose RMS key is being deleted.

Format 4 characters

SKEY category

The SKEY category includes the following actions:

- ADDing a secondary key (see “Adding a secondary key” on page 237)
- MODifying a secondary key (see “Modifying a secondary key” on page 240)
- CONnecting a secondary key (see “Connecting a secondary key” on page 242)
- DISconnecting a secondary key (see “Disconnecting a secondary key” on page 243)
- DELeting a secondary key (see “Deleting a secondary key” on page 244)

For command syntax, refer to the appropriate section.

Adding a secondary key

To add a secondary key, use the following syntax:

SKEY ADD NAME=*secondary-key-name* DSET=*data-set-name*

INDEX=*index-name* [UNIQUE=*option*] [DIRECTION=*option*]

[POINTER=*option*] [PTYPE=*option*] [DATATYPE=*option*]

[DUPS=*nnn*]

NAME=*secondary-key-name*

Description *Required*. Specifies the name of the secondary key to be added to the specified index and data set.

Format SK*xx*

DSET=*data-set-name*

Description *Required*. Specifies the name of the data set to which the index for this secondary key belongs.

Format 4 characters

INDEX=*index-name*

Description *Required*. Specifies the name of the index to which the secondary key is being added.

Format IX*xx*

UNIQUE=*option*

Description *Optional*. Specifies whether the secondary key must be unique.

Default N

Options

Y	Yes
N	No

DIRECTION=*option*

Description *Optional*. Specifies the direction in which the keys are sorted in the index file.

Default F

Options F Forward

R Reverse

B Both

POINTER=*option*

Description *Optional*. Specifies whether to use pointers to ensure that records with identical keys are retrieved in the order in which they occur in the data set file.

Default N

Options Y Yes

N No

PTYPE=*option*

Description *Optional*. Specifies the pointer type stored with the secondary key in order to ensure that records with identical keys are retrieved in the order in which they occur in the file.

Default D

Options D Direct; use a control key

I Indirect; use a relative record number (RRN)

DATATYPE=*option*

Description *Optional*. Specifies whether the secondary keys are to be sorted according to data type rather than treating all secondary keys as character strings.

Default N

Options Y Yes; sort secondary keys by data type

N No; sort secondary keys as character strings

DUPS=*nnn*

Description *Optional*. Specifies the percentage of duplicate values you expect for this secondary key.

Default 5

Options 0–99 (up to 99%)

Modifying a secondary key

To modify a secondary key, use the following syntax:

SKEY MOD NAME=*secondary-key-name* DSET=*data-set-name*

[UNIQUE=*option*] [DIRECTION=*option*] [POINTER=*option*]

[PTYPE=*option*] [DATATYPE=*option*] [DUPS=*nnn*]

NAME=*secondary-key-name*

Description *Required*. Specifies the name of the secondary key to be modified.

Format SK*xx*

DSET=*data-set-name*

Description *Required*. Specifies the name of the data set associated with the index to which the secondary key being modified belongs.

Format 4 characters

UNIQUE=*option*

Description *Optional*. Specifies whether the secondary key must be unique.

Default N

Options

Y	Yes
N	No

DIRECTION=*option*

Description *Optional*. Specifies the direction in which the keys are sorted in the index file.

Default F

Options

F	Forward
R	Reverse
B	Both

POINTER=*option*

Description *Optional*. Specifies whether to use pointers to ensure that records with identical keys are retrieved in the order in which they occur in the data set file.

Default N

Options Y Yes

N No

PTYPE=*option*

Description *Optional*. Specifies the pointer type stored with the secondary key in order to ensure that records with identical keys are retrieved in the order in which they occur in the file.

Default D

Options D Direct; use a control key

I Indirect; use a relative record number (RRN)

DATATYPE=*option*

Description *Optional*. Specifies whether the secondary keys are to be sorted according to data type rather than treating all secondary keys as character strings.

Default N

Options Y Yes; sort secondary keys by data type

N No; sort secondary keys as character strings

DUPS=*nnn*

Description *Optional*. Specifies the percentage of duplicate values you expect for this secondary key

Default 5

Options 0–99 (up to 99%)

Connecting a secondary key

To connect a secondary key, use the following syntax:

SKEY CON NAME=*secondary-key-name* DSET=*data-set-name*

DITM=*data-item-name*

NAME=*secondary-key-name*

Description *Required*. Specifies the name of the secondary key to which to connect the specified data item of the specified data set.

Format SK*xx*

DSET=*data-set-name*

Description *Required*. Specifies the name of the data set whose secondary key is being appended.

Format 4 characters

DITM=*data-item-name*

Description *Required*. Specifies the name of the data item to connect to the secondary key.

Format 4 characters

Consideration New items are connected at the end of the list of data items for the secondary key.

Disconnecting a secondary key

To disconnect a secondary key, use the following syntax:

SKEY DIS NAME=*secondary-key-name* DSET=*data-set-name*

DITM=*data-item-name*

NAME=*secondary-key-name*

Description *Required*. Specifies the name of the secondary key that is having a data item removed from it.

Format SK*xx*

DSET=*data-set-name*

Description *Required*. Specifies the name of the data set associated with the index to which the secondary key belongs.

Format 4 characters

DITM=*data-item-name*

Description *Required*. Specifies the name of the data item to disconnect from the secondary key.

Format 4 characters

Deleting a secondary key

To delete a secondary key, use the following syntax:

SKEY DEL NAME=*secondary-key-name* DSET=*data-set- name*

NAME=*secondary-key-name*

Description *Required*. Specifies the name of the secondary key to be deleted.

Format 4 characters

DSET=*data-set- name*

Description *Required*. Specifies the name of the data set associated with the index whose secondary key is being deleted.

Format 4 characters

Consideration All data items in the secondary key must be disconnected first.

SLOG category

The SLOG category includes the following actions:

- ADDing a system log (see “Adding a system log” on page 246)
- MODifying a system log (see “Modifying a system log” on page 248)
- DELeting a system log (see “Deleting a system log” on page 250)

For command syntax, refer to the appropriate section.

Adding a system log

To add a system log, use the following syntax:

```
SLOG ADD DBDESC=database-name BLOCKS=number-of-blocks
  FSPEC1=system-log-file-1-spec FSPEC2=system-log-file-2-spec
  [BLKSIZ=size]
```

DBDESC=*database-name*

Description *Required*. Specifies the name of the database to which the system logs will belong.

Format 6 characters

BLOCKS=*number-of-blocks*

Description *Required*. Specifies the total number of logical blocks in the system log.

Options 5–64K

FSPEC1=*system-log-file-1-spec*

Description *Required*. Specifies the physical file specification for the initial system log.

Format 1–44 alphanumeric characters. Must represent a valid VMS file specification.

Consideration If the file-spec contains special characters, it must be enclosed in double quotes (").

FSPEC2=*system-log-file-2-spec*

Description *Required*. Specifies the physical file specification for the secondary system log.

Format 1–44 alphanumeric characters. Must represent a valid VMS file specification.

Consideration If the file-spec contains special characters, it must be enclosed in double quotes (").

BLKSIZ=*size*

Description *Optional*. Specifies the size of the system log blocks in multiples of 1024 bytes.

Default 1

Options 1–10

Consideration The DBA utility will adjust this value during validation.

Modifying a system log

To modify a system log, use the following syntax:

SLOG MOD DBDESC=*database-name* BLOCKS=*number-of-blocks*

FSPEC_1=*system-log-file-1-spec* FSPEC_2=*system-log-file-2-spec*

[BLKSIZ=*size*]

DBDESC=*database-name*

Description *Required*. Specifies the name of the database to which the system logs belong.

Format 6 characters

BLOCKS=*number-of-blocks*

Description *Required*. Specifies the total number of logical blocks in the system log.

Options 5–64K

FSPEC_1=*system-log-file-1-spec*

Description *Required*. Specifies the physical file specification for the initial system log.

Format 1–44 alphanumeric characters. Must represent a valid VMS file specification.

Consideration If the file-spec contains special characters, it must be enclosed in double quotes (").

FSPEC_2=*system-log-file-2-spec*

Description *Required*. Specifies the physical file specification for the secondary system log.

Format 1–44 alphanumeric characters. Must represent a valid VMS file specification.

Consideration If the file-spec contains special characters, it must be enclosed in double quotes (").

BLKSIZ=*size*

Description *Optional*. Specifies the size of the system log blocks in multiples of 1024 bytes.

Default 1

Options 1–10

Consideration The DBA utility will adjust this value during validation.

Deleting a system log

To delete a system log, use the following syntax:

SLOG DEL DBDESC=*database-name*

DBDESC=*database-name*

Description *Required*. Specifies the name of the database whose system log is to be deleted.

Format 6 characters

TLOG category

The TLOG category includes the following actions:

- ADDing a task log (see “Adding a task log” on page 251)
- MODifying a task log (see “Modifying a task log” on page 253)
- DELeting a task log (see “Deleting a task log” on page 254)

For command syntax, refer to the appropriate section.

Adding a task log

To add a task log, use the following syntax:

TLOG ADD DBDESC=*database-name* BLOCKS=*number-of-blocks*

FSPEC=*task-log-file-spec* [BUFFERS=*no.-of-buffers*] [BLKSIZ=*size*]

DBDESC=*database-name*

Description *Required*. Specifies the name of the database to which the task log will belong.

Format 6 characters

BLOCKS=*number-of-blocks*

Description *Required*. Specifies the total number of logical blocks in the task log.

Format 1–9 numeric characters

FSPEC=*task-log-file-spec*

Description *Required*. Specifies the physical file specification for the task log.

Format 1–44 alphanumeric characters. Must represent a valid VMS file specification.

Consideration If the file-spec contains special characters, it must be enclosed in double quotes (").

BUFFERS=*no.-of-buffers*

Description *Optional*. Specifies the number of copies of the task log file.

Default 5

Options 1–32

BLKSIZ=*size*

Description *Optional*. Specifies the size of the task log blocks in multiples of 2048 bytes.

Default 1

Options 1–10

Consideration The DBA utility will adjust this value during validation.

Modifying a task log

To modify a task log, use the following syntax:

TLOG MOD DBDESC=*database-name* BLOCKS=*number-of-blocks*
FSPEC=*task-log-file-spec* [BUFFERS=*no.-of-buffers*]
[BLKSIZ=*size*]

DBDESC=*database-name*

Description *Required*. Specifies the name of the database whose task log is being modified.

Format 6 characters

BLOCKS=*number-of-blocks*

Description *Required*. Specifies the total number of logical blocks in the task log.

Format 1–9 numeric characters

FSPEC=*task-log-file-spec*

Description *Required*. Specifies the physical file specification for the task log.

Format 1–44 alphanumeric characters. Must represent a valid VMS file specification.

Consideration If the file-spec contains special characters, it must be enclosed in double quotes (").

BUFFERS=*no.-of-buffers*

Description *Optional*. Specifies the number of copies of the task log file.

Options 1–32

BLKSIZ=*size*

Description *Optional*. Specifies the size of the task log blocks in multiples of 2048 bytes.

Default 1

Options 1–10

Consideration The DBA utility will adjust this value during validation.

Deleting a task log

To delete a task log, use the following syntax:

TLOG DEL DBDESC=*database-name*

DBDESC=*database-name*

Description *Required*. Specifies the name of the database whose task log is being deleted.

Format 6 characters

USER category

The USER category includes the following actions:

- ADDing a SUPRA PDM user (see “Adding a user” on page 256)
- MODifying a SUPRA PDM user (see “Modifying a user” on page 257)
- DELeting a SUPRA PDM user (see “Deleting a user” on page 258)

For command syntax, refer to the appropriate section.

Adding a user

To add a SUPRA PDM user, use the following syntax:

USER ADD NAME=*user-name* [AUTH=*option*]

  [PASSWORD=*password*]

NAME=*user-name*

Description *Required*. Specifies the SUPRA PDM user name to be added.

Format 1–30 characters

Considerations

- The SUPRA PDM user DATA-DICTIONARY may not be changed by DIRM.
- DIRM attempts to keep the users in alphabetical order.

AUTH=*option*

Description *Optional*. Specifies the type of access available to this user.

Default DP

Options

PRIV	Privileged; access to all users and functions.
DA	DBA/Utilities; access to all users and functions except privileged.
DP	Development; access to database, data set, read-only access to views, logical data items, and program functions.
READ	Read-only; read-only access to databases and data sets.
RDM	RDM user with no access to DBA functions; defined for use by RDML programs.

PASSWORD=*password*

Description *Optional*. Specifies the password for this SUPRA PDM user.

Default Blank

Format 1–8 characters

Modifying a user

To modify a SUPRA PDM user, use the following syntax:

USER MOD NAME=*user-name* [AUTH=*option*]

[PASSWORD=*password*]

NAME=*user-name*

Description *Required*. Specifies the SUPRA PDM user name to be added.

Format 1–30 characters

Considerations

- The SUPRA PDM user DATA-DICTIONARY may not be changed by DIRM.
- DIRM attempts to keep the users in alphabetical order.

AUTH=*option*

Description *Optional*. Specifies the type of access available to this user.

Default DP

Options

PRIV	Privileged; access to all users and functions.
DA	DBA/Utilities; access to all users and functions except privileged.
DP	Development; access to database, data set, read-only access to views, logical data items, and program functions.
READ	Read-only; read-only access to databases and data sets.
RDM	RDM user no access to DBA functions; defined for use by RDML programs.

PASSWORD=*password*

Description *Optional*. Specifies the password for this SUPRA PDM user.

Default Blank

Format 1–8 characters

Deleting a user

To delete a SUPRA PDM user, use the following syntax:

USER DEL NAME=*user-name*

NAME=*user-name*

Description *Required*. Specifies the SUPRA PDM user to be deleted.

Format 1–30 characters

VTBL category

The VTBL category includes the following actions:

- ADDing a validation table (see “Adding a validation table” below)
- MODifying a validation table (see “Modifying a validation table” below)
- DELeting a validation table (see “Deleting a validation table” on page 260)

For command syntax, refer to the appropriate section.

Adding a validation table

To add a validation table, use the following syntax:

VTBL ADD NAME=*table-name*

NAME=*table-name*

Description *Required*. Specifies the name of the validation table to be added.

Format 1–30 characters

Modifying a validation table

To modify a validation table, use the following syntax:

VTBL MOD NAME=*table-name*

NAME=*table-name*

Description *Required*. Specifies the name of the validation table to be modified.

Format 1–30 characters

Consideration VTBL MOD deletes all of the existing entries for the table and then rebuilds the table using all of the following input lines until a line starting with 'END-TABLE.' is met. If you want to modify comments here, the COMMENT-MOD command must follow the END-TABLE. END-TABLE is always required after a VTBL MOD.

Deleting a validation table

To delete a validation table, use the following syntax:

VTBL DEL NAME=*table-name*

NAME=*table-name*

Description *Required*. Specifies the name of the validation table to be deleted.

Format 1–30 characters

Batch DIRM examples

There are two logicals and a file that need defining and creating before you can use the command procedure in batch. They are:

MANTIS_INPUT defined to point to a file with the following lines of information:

```
{ECHO ON}
{1}
{}
{CANCEL}
```

NOTE

The 1 in the brackets indicates for this example file that a LOAD command is being requested to be processed by DIRM. You may change that value to correspond to the value of the function in DIRM that you will be using.

CSIDIRM_KNOWNGOODBACKUP is defined to TRUE. As with interactive DIRM if you do not answer affirmative to having a good backup DRIM will not process your requests and will return out of the procedure so that you may first create a good backup.

The following is an example of a DIRM Batch job:

```
$ ! This .COM file runs Batch Dirm
$
$ define dirm_input disk:[dir.subdir]file_part.dirm_input ! input
  file
$ dirm
$
$       if dirm_exit .eqs. "OK" then goto dirm_run_ok
$
$       type sys$input:
=========================
=== ERRORS in DIRM run  ====
=========================
$
$       exit
$
$ dirm_run_ok:
$
$       type sys$input:
=========================
===== DIRM run OK         =====
=========================
$
$       exit
```

The following is an example of a DIRM input file, FILE_PART.DDL:

```
! This is a DIRM input file containing modifications to the
! PART file. All new modifications are added at the end and are
! activated using a GOTO, so this growing file will always
  contain
! ALL of our modifications made to the PART file.
!
: PART file modifications.
:
option mod, echo=c.
!
option mod, goto=921009.
!
:910101:
!
: First change, add field NEW1, January 1991
:
ditm add, dset=part, name=new1, type=c, size=8.
!
:921009:
!
: Second change, add field NEW2, October 1992
:
ditm add, dset=part, name=new2, after=new1, size=4, type=n.
!
end.
```

5

Using the Database Verify utility

Database Verify utility introduction

You use the Database Verify utility to:

- Verify a data set’s physical record locations and record pointers
- Verify a data set’s physical characteristics with the information in the database description file and the information on the SUPRAD Directory
- Gather physical statistics about a data set

NOTE

See the following page for specific information about which the Database Verify utility verifies and gathers statistics.

What to do before you run the Database Verify utility

Before you run the Database Verify utility, be sure you have sufficient resources (see “Determining VMS resource requirements for the Database Verify utility” on page 266 for UNIX requirements or “Determining UNIX resource requirements for the Database Verify utility” on page 268 for VMS requirements).

Reasons you might use the Database Verify utility

You could use the Database Verify utility to gather data set statistics and/or to determine if a data set requires restructuring, expansion or repair. For example, you may run the Database Verify utility if you had a recovery failure or if you encounter integrity errors. You could also use the Database Verify utility to determine if you need to recompile your database description.

Specific information that the Database Verify utility verifies and/or gathers statistics about

The Database Verify utility quickly and efficiently verifies:

- A data set’s physical and logical size
- Related data set linkpath next and previous pointers
- Related data set record codes
- Primary data set’s linkpath pointers
- Primary data set’s synonym next and previous pointers
- Primary and related data set’s keys

The Database Verify utility may also be used to obtain statistics about:

- Current record density of a file
- Average number of records in a related data set linkpath
- Longest, shortest, and average length of a related data set
- Number of related linkpaths fragmented across blocks and control interval boundaries
- Number of primary keys with and without related data set records
- Number of primary data set synonyms
- Longest and average primary data set’s synonym chain length
- Number of primary data set’s synonyms out of block

How the Database Verify utility works

When you run the Database Verify utility, it performs the following steps:

1. Examines the existing, compiled database-description file for any physical and logical details that were not specified on the command line.

2. Checks that the details in the database description file match the details of the database held on the SUPRA Directory.

3. Verifies the physical location and the logical relationships of records and record codes with a data set’s associated primary and related data set keys.

4. Gathers statistical information about a data set.

5. Prints a detailed report of errors found and/or the statistical information about a data set.

Determining VMS resource requirements for the Database Verify utility

The Database Verify utility uses the DEC SORT facility to create sort work files for processing. This section provides information about the DEC SORT facility and calculations to determine the space you need for the sort work files.

The DEC SORT facility

The Database Verify utility uses the DEC SORT facility to create work files into which the facility returns sorted records. The Database Verify utility uses these records so that a data set's logical and physical record locations can be validated and statistical information about a data set can be compiled. Before you run the Database Verify utility, you must plan your disk space requirements to allow enough free space for the sort work files.

When processing very large data sets, you may not have enough disk space for the sort work files. If the DEC SORT facility cannot extend its work files because of lack of disk space, the Database Verify utility aborts. If this happens, allocate additional sort work files and start again. You do not have to restore your database files from a backup or run a recovery operation. The Database Verify utility performs all of its functions in a read-only manner.

VMS allows up to ten work files: SORTWORK0 through SORTWORK9. By default, two work files are created in the directory assigned the logical name SYS$SCRATCH. You may specify alternative logical assignments using the VMS define command as follows:

```
  $ DEFINE SORTWORKn device:[directory]filename.type
```

where *n* is a digit between 0 and 9, inclusive.

The higher the sort work file you specify determines how many sort work files the Database Verify utility uses. For example, if you define SORTWORK4, then five work files are used.

Calculating sort work file size for VMS

Before using the Database Verify utility, ensure you have enough disk space for all of your sort work files. The Database Verify utility cannot recover if DEC SORT is unable to extend its work files as necessary. The free disk space needed for the sort files for primary and related data sets is approximately as follows:

Data set type	**Disk space requirements**
Primary data sets	(16 bytes + primary key length) * total logical records
Related data sets	((16 bytes for each linkpath) + (related key length for each linkpath)) * total logical records

Determining UNIX resource requirements for the Database Verify utility

The Database Verify utility uses a customized sort facility to create sort work files for processing. This section provides information about the customized sort facility, optimizing space for the customized sort facility and calculations to determine the disk space you need for the sort work files.

The customized sort facility for UNIX

The Database Verify utility uses a customized sort facility to create work files into which the facility returns sorted records. The Database Verify utility uses these records so that a data set's logical and physical record locations can be validated and statistical information about a data set can be compiled. Before you run the Database Verify utility, you must plan your disk space requirements to allow enough free space for the sort work files and a temporary file containing the fully sorted records. The size of the sort work files are approximately the same as the temporary work files. Sort work files are deleted by the sort facility once the temporary work files have been built. Temporary work files are deleted by the Database Verify utility once the utility has completed its validation and statistics report.

Number of sort work files used and temporary files created by the customized sort facility

The number of sort work files used and the number of temporary work files created depends on the type of data sets and the number of linkpaths present. For primary data sets, one sort work file and one temporary work file is created. For related data sets, one sort work file and one temporary work file is created for each linkpath. The Database Verify utility permits up to 32 sort work files and temporary work files (accommodating up to 32 linkpaths) per data set.

Where the customized sort facility acquires space

The customized sort facility acquires space for the sort work files from the / directory while it acquires space for the temporary work files from the current working directory. You can direct the sort work files elsewhere, but you may not redirect the temporary work files (see “Controlling the placement of sort work files” under “Optimizing space for the customized sort work facility” on page 270).

What happens if the customized sort facility does not have enough space

When processing very large data sets, you may not have enough disk space for the sort work files or the temporary work files. If the sort facility cannot extend its work files because of lack of disk space, a UNIX I/O error is reported and the Database Verify utility aborts. If this happens, you must clean up your current file system, or locate a different working directory with additional space in it. You may then run the Database Verify utility again. You do not have to restore your database files from a backup or run a recovery operation. The Database Verify utility performs all of its functions in a read-only manner.

Optimizing space for the customized sort work facility

You may control the efficiency of the customized sort facility by balancing memory against I/O activity. It is possible to control the maximum number of records held in memory by the sort facility before it begins placing records in a sort work file. It is also possible to specify where those files will be placed.

The number and size of records held in memory has the most impact on the length of time the Database Verify utility takes to run. For large data sets with many linkpaths or if many data sets are to be verified in the same run, keeping too few records in memory generally causes slower sort performance due to physical I/O to the sort work files. However, keeping too many records in memory can cause excessive UNIX system thrashing or job failure due to excessive real memory requirements.

To determine an optimum memory-I/O balance for your situation, you must first determine how many records to maintain in memory before writing to sort work files. For small data sets on a lightly loaded machine, the customized sort facility default (32,768 records allocated for sort work files) should be sufficient. For much larger data sets running on a heavily loaded machine, you will have to make some calculations to find an optimal time/disk/memory value.

Determining the amount of records to maintain in memory. To determine the total number of records you should maintain in memory before writing to a work file, perform the following:

1. Use the UNIX sar and vmstat commands to determine the average amount of memory available.

2. Take the average amount of memory available (from step 1) and multiply this amount by 80% to obtain a realistic amount of available memory.

3. Determine the size of the records to be sorted using the following calculation:

   ```
   sort record size (for primary and related data sets)=
   16 + length of a key
   ```

   Keep in mind that a related data set can have more than one key; therefore, it can have more than one sort work file. For a related data set, one sort work file is allocated for each key (corresponding to each linkpath).

4. Divide the amount of available memory (from step 2) by the total size of the records to be sorted (from step 3). The resulting value represents the total number of records you should maintain in memory before writing to a sort work file.

Controlling the placement of sort work files to increase sort efficiency. An I/O is dispatched to a sort work file when there are more input records than the number held in memory. When all of the records are distributed to various sort work files, these files and any records held in memory are merged together to produce a temporary work file.

You can control where the sort work files are placed and their sizes and direct them to multiple file systems. This will increase the speed of the sort as the sort work files are merged in parallel.

You use an environment variable called COSORT_TUNER to set the locations and sizes of the sort work files.

COSORT_TUNER is an input parameter for a sorting product called COSORT™. COSORT™ is a trademark of Innovative Routines, International, New York, USA.

You specify COSORT_TUNER values by issuing the following UNIX command. It is unnecessary to issue this command if all of the default values (listed below) are acceptable. You can specify values in the following manner: gigabytes (GB), megabytes (MB), kilobytes (K), bytes (30,000), or decimal equivalents (30K).

```
COSORT_TUNER=[block-size][ _max-memory][::work-areas];export
  COSORT_TUNER
```

block-size

Description	*Optional*. Specifies the number of kilobytes in a block of records to be used for the sort.
Default	80K
Options	32K–500K

_max-memory

Description *Optional*. Specifies the maximum amount of memory (in megabytes) to be used for the sort.

Default 20 MB

Options 6 MB up to the maximum available memory

Consideration If specified, this value must be preceded by an underscore (_).

::*work-areas*

Description *Optional*. Specifies one or more overflow directories for distributing input records for the sort. You can assign a maximum byte limitation for each directory as follows:

```
::dir1[=val1]:dir2[=val2]…
```

Format directory name: Valid pathname

Defaults directory name: /

byte-limitation value: infinite (The sort process will write until it runs out of space.)

Options Effective size of a disk file (actual usable space given the local physical disk space and operating system constraints)

Considerations

- If specified, the first work area must be preceded by two colons (::). Succeeding directory/value specifications must be separated by single colons (:).
- When the number of input records exceeds the _max-memory parameter, the sort process will distribute these records to a sort work file. The process will create and fill a succession of files in the overflow directories you have specified in the sequence you have specified. This sequence is repeated for successive files (if you specified dir1:dir2, the input records would be distributed as follows: file1 to dir1, file2 to dir2, file3 to dir1, file4 to dir2, etc.) If the specified directories become full, any additional data will be sent to the default / directory.

General consideration

The COSORT_TUNER command must end with the ;export COSORT_TUNER clause. This clause places the COSORT_TUNER variable in the current UNIX environment.

Example

The following example reads and writes blocks of 500K. 25 MB of RAM has been authorized for sorting purposes. Overflow records will be sent alternately to /tmp1 (using the default of all available space) and to /tmp2 (for a maximum of 500 MB). If these two directories become full, any additional data will be sent to the default / directory.

```
COSORT_TUNER=500kb_25mb::/tmp1:/tmp2=500mb;export COSORT_TUNER
```

Calculating sort work file size for UNIX

Before you use the Database Verify utility, ensure you have enough disk space for all of your sort work files and temporary work files. The Database Verify utility cannot recover if the sort facility is unable to extend its work files as necessary. The free disk space needed for the sort work files and the temporary work file for primary and related data sets is approximately as follows:

Data set type	Disk space requirements
Primary data sets	(16 bytes + primary key length) * total logical records
Related data sets	((16 bytes for each linkpath) + (related key length for each linkpath)) * total logical records

Running the Database Verify utility

You run the Database Verify utility using the CSIDBVER command (VMS) or the csiuxdbver command (UNIX). This section contains information about:

- VMS CSIDBVER.CLD file
- Database Verify utility command parameters
- How to run the Database Verify utility using the following methods:
  - Executing using a verify list file (to verify more than one data set at a time)
  - Executing interactively at the command level (VMS) or shell (UNIX)
  - Spawning as a subprocess (VMS) or executing as a background task (UNIX)
  - Executing using a command file (VMS) or script (UNIX)

CSIDBVER.CLD file (VMS only)

The CSIDBVER command is described to VMS in a file called CSIDBVER.CLD. You add this file to the system command table at boot-time or to your user command table at login time as follows:

```
$SET COMMAND SUPRA_COMS:CSIDBVER.CLD
```

You usually carry out this procedure when you install SUPRA.

Database Verify utility command parameters

When you run the Database Verify utility at the command line, it takes a series of parameters. The parameters are classified as follows:

- **Data set parameters.** Specifies the name of the data set to be verified and, optionally, whether or not statistics are to be compiled for the data set:

VMS

```
/DATASET_NAME=data-set-name

/STATISTICS
```

UNIX

```
DATASET_NAME=data-set-name

STATISTICS
```

- **Database parameters**. Specifies the name of the database and the database password containing the data set to be verified:

VMS

```
database-name

/PASSWORD=database-password
```

UNIX

```
DB_NAME=database-name

PASSWORD=database-password
```

- **Directory access parameter**. Specifies that the Database Verify utility either:

  - Validates the information in the SUPRAD Directory with the information in the database description file ONLY. (It does not validate the physical data file with the information in the SUPRAD Directory.)

  - Does not validate the data set information in the SUPRAD Directory with the information in the database description file. (However, it will validate the information in the physical data file with the information in the database description file.)

VMS

```
/DIRECTORY=option
```

UNIX

```
DIRECTORY=option
```

- **Change file parameter**. Specifies a physical file that contains a list of data set names to be verified. Use this parameter if you wish to verify more than one data set at a time.

VMS

```
/CHANGE_LIST=change-file-spec
```

UNIX

```
VERIFY_LIST=change-file-path
```

NOTE

The terms “change file” and “verify list file” are used interchangeably on the following pages, but both refer to the file containing the list of data set names to be verified.

The following are the Database Verify utility parameters and their descriptions:

CSIDBVER VMS

csiuxdbver UNIX

Description *Required*. The command that initiates the Database Verify utility.

Format Enter exactly as shown.

database-name VMS

DB_NAME=*database-name* UNIX

Description *Required*. Specifies the name of the compiled database description in the SUPRAD Directory that contains the data set to be verified.

Format 6 alphanumeric characters

Considerations

- The database must exist in the SUPRAD Directory.
- A logical name for the database must exist.
- A compiled database description file must exist.

/PASSWORD=*database-password* VMS

PASSWORD=*database-password* UNIX

Description *Required* only if the database is protected by a password. Specifies the password assigned to the database.

Considerations

- If the database description has a password, the password you enter must match the database description password. If you enter an invalid password, the Database Verify utility terminates and returns you to the command level (VMS) or the shell (UNIX).
- The password is displayed on the screen as you type. To avoid displaying the password, omit the /PASSWORD (no slash required under UNIX) qualifier. The Database Verify utility prompts you to enter the password and does not display what you enter.
- If you direct the output of the Database Verify utility to a file, the system will wait until you enter a password although it does not actually show a prompt that says 'Password:'.

/DATASET_NAME=*data-set-name* VMS

DATASET_NAME=*data-set-name* UNIX

Description *Required*. Specifies the name of the data set to be validated.

Format 4 alphanumeric characters

Consideration The specified data set must exist in the database.

/STATISTICS VMS

STATISTICS UNIX

Description *Required*. Specifies if statistics are to be compiled for the data set while it is being verified.

Format Enter exactly as shown.

Consideration Statistics will not be compiled if the parameter is not specified.

/DIRECTORY=*option* VMS

DIRECTORY=*option* UNIX

Description *Optional*. Specifies that you want either one of the following: (1) the data set information in the SUPRAD Directory validated with the information in the database description file ONLY (the information in the Directory is not validated with the physical file) or (2) that you do not want the data set information in the SUPRAD Directory validated with the data set information in the database description file.

Format 2–4 alphabetic characters

Options

ONLY	Validates the data set information in the SUPRAD Directory with that in the database description file ONLY (it does not validate data set information in the SUPRAD Directory with the data set physical file).
NO	Does not validate the data set information in the SUPRAD Directory with that in the database description file.

Consideration If you do not specify a DIRECTORY parameter, the Database Verify utility will normally validate the information in the data set physical file with the data set information in both the database description file and the SUPRAD Directory.

/CHANGE_LIST=*change-file-spec* VMS

VERIFY_LIST=*change-file-path* UNIX

Description *Optional*. Specifies a file containing a list of data sets to be verified. Optionally, the STATISTICS parameter can be added to each data set in the list for which statistics are to be complied during verification. Use this parameter if you wish to verify more than one file at a time.

Using the Database Verify utility with a verify list file

You can verify multiple data sets in a single run by specifying a change file on the command line. To create and use a change file, follow these steps:

1. Create a verify list file using a standard text editor. The example below will verify and produce statistics for several files. Lines beginning with an exclamation mark (VMS) are comments (comment lines are not permitted in a verify list file under UNIX).

VMS

```
! Change List for verifying several files for database

! CUSTDB

!           VMS EXAMPLE filename=flist.dat

/DATASET_NAME=CUST/STATISTICS

/DATASET_NAME=PROD

/DATASET_NAME=ORDR/STATISTICS
```

In this example, statistics will be generated for cust and ordr, but not for prod.

UNIX

```
DATASET_NAME=CUST STATISTICS

DATASET_NAME=PROD

DATASET_NAME=ORDR STATISTICS
```

In this example, statistics will be generated for cust and ordr, but not for prod.

2. Do one of the following:

VMS

Invoke CSIDBVER with the change file in the format:

```
$CSIDBVER/CHANGE_LIST=FLIST.DAT CUSTDB
```

UNIX

Invoke csiuxdbver with the change file in the format:

```
csiuxdbver db_name=custdb verify_list=/usr/acct/flist.dat
```

NOTE

You can also direct the output to a file when you use a change file.

NOTE

You may include only data set and statistic parameters in the file; you cannot include the DIRECTORY, PASSWORD, and DB_NAME parameters. Specify these parameters on the command line or in a command file or a script if you use one.

UNIX

If you forget PASSWORD and you direct the output to a file, the Database Verify utility will hang. Simply enter the password, and the utility will continue.

Executing the Database Verify utility interactively

To execute the Database Verify utility interactively, enter the command with all parameters on the command line. The following shows some of the ways you can run the Database Verify utility:

VMS

```
$CSIDBVER CUSTDB/DATASET_NAME=CUST/DIRECTORY=NO/STATISTICS
```

UNIX

```
csiuxdbver db_name=custdb dataset_name=cust directory=no statistics
```

The above examples will verify and compile statistics for data set CUST, but it will not validate the information about CUST in the SUPRAD Directory.

Spawning the Database Verify utility as a subprocess (VMS) or as a background process (UNIX)

VMS

To spawn a Database Verify utility as a subprocess, type SPAWN at the command line together with a list of qualifiers and the database-name parameter. For example, the following spawns CSIDBVER as a subprocess:

```
$SPAWN CSIDBVER CUSTDB/DATASET_NAME=CUST/STATISTICS
```

UNIX

To submit the Database Verify utility as a background process, enter the command csiuxdbver with a list of parameters. For example, the following will run the csiuxdbver image as a background process and place the report in the current directory in a file named csiuxdbver.out:

```
nohup csiuxdbver db_name=custdb dataset_name=cust statistics/

>csiuxdbver.out 2>>csiuxdbver.out &
```

Executing a Database Verify command file (VMS) or script (UNIX)

The following is an example VMS command file and UNIX script DO_VERIFY.COM to take statistics for a data set named CUST. You may create the command file or script with any text editor.

VMS

```
$IF P1 .EQS. "" THEN EXIT
$IF P2 .EQS. "" THEN EXIT
$CSIDBVER 'P1'/DATASET_NAME='P2'/STATISTICS
$CSIDBVER 'P1'/DATASET_NAME='P2'/STATISTICS
$WRITE SYS$OUTPUT "CSIDBVER COMPLETED"
$EXIT
```

UNIX

```
#
if test "$1" = ""
then
    echo 'no database name supplied';
else
if test "$2" = ""
then
    echo 'no data set name supplied';
else
csiuxdbver db_name=$1 data-set-name=$2 statistics
echo 'Verification of file' $2 'is complete'
fi
fi
#
```

Executing a Database Verify utility command file (VMS) or script (UNIX) online

To execute the command file online, enter:

VMS

```
@DO_VERIFY.COM CUSTDB CUST
```

UNIX

```
chmod 744 do_verify.com; do_verify.com custdb cust
```

You may also specify a CHANGE_LIST in the command file or script in place of the data set name.

Submitting a Database Verify utility command file to batch (VMS) or script as a background process (UNIX)

You may submit a database verify command file or script by entering

VMS

```
$ SUBMIT/PARAM=(CUSTDB,CUST) DO_VERIFY.COM
```

UNIX

```
nohup do_verify.com custdb cust >verify.log 2>>verify.log &
```

The UNIX example runs the do_verify.com script as a background process and directs the output to the current directory in a file named verify.log.

Index